

Xiao Bai · Edwin R. Hancock  
Tin Kam Ho · Richard C. Wilson  
Battista Biggio · Antonio Robles-Kelly (Eds.)

LNCS 11004

# Structural, Syntactic, and Statistical Pattern Recognition

Joint IAPR International Workshop, S+SSPR 2018  
Beijing, China, August 17–19, 2018  
Proceedings

 Springer

EXTRAS ONLINE

*Commenced Publication in 1973*

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

## Editorial Board

David Hutchison

*Lancaster University, Lancaster, UK*

Takeo Kanade

*Carnegie Mellon University, Pittsburgh, PA, USA*

Josef Kittler

*University of Surrey, Guildford, UK*

Jon M. Kleinberg

*Cornell University, Ithaca, NY, USA*

Friedemann Mattern

*ETH Zurich, Zurich, Switzerland*

John C. Mitchell

*Stanford University, Stanford, CA, USA*

Moni Naor

*Weizmann Institute of Science, Rehovot, Israel*

C. Pandu Rangan

*Indian Institute of Technology Madras, Chennai, India*

Bernhard Steffen

*TU Dortmund University, Dortmund, Germany*

Demetri Terzopoulos

*University of California, Los Angeles, CA, USA*

Doug Tygar

*University of California, Berkeley, CA, USA*

Gerhard Weikum

*Max Planck Institute for Informatics, Saarbrücken, Germany*



More information about this series at <http://www.springer.com/series/7412>

Xiao Bai · Edwin R. Hancock  
Tin Kam Ho · Richard C. Wilson  
Battista Biggio · Antonio Robles-Kelly (Eds.)

# Structural, Syntactic, and Statistical Pattern Recognition

Joint IAPR International Workshop, S+SSPR 2018  
Beijing, China, August 17–19, 2018  
Proceedings

*Editors*

Xiao Bai  
Beihang University  
Beijing  
China

Edwin R. Hancock  
University of York  
York  
UK

Tin Kam Ho  
IBM Research – Thomas J. Watson  
Research  
Yorktown Heights, NY  
USA

Richard C. Wilson  
University of York  
Heslington, York  
UK

Battista Biggio  
University of Cagliari  
Cagliari  
Italy

Antonio Robles-Kelly  
Data 61 - CSIRO  
Canberra, ACT  
Australia

ISSN 0302-9743                      ISSN 1611-3349 (electronic)  
Lecture Notes in Computer Science  
ISBN 978-3-319-97784-3              ISBN 978-3-319-97785-0 (eBook)  
<https://doi.org/10.1007/978-3-319-97785-0>

Library of Congress Control Number: 2018950098

LNCS Sublibrary: SL6 – Image Processing, Computer Vision, Pattern Recognition, and Graphics

© Springer Nature Switzerland AG 2018

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG  
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

# Preface

This volume contains the papers presented at the joint IAPR International Workshops on Structural and Syntactic Pattern Recognition (SSPR 2018) and Statistical Techniques in Pattern Recognition (SPR 2018). S+SSPR 2018 was jointly organized by Technical Committee 1 (Statistical Pattern Recognition Technique, chaired by Battista Biggio) and Technical Committee 2 (Structural and Syntactical Pattern Recognition, chaired by Antonio Robles-Kelly) of the International Association of Pattern Recognition (IAPR). It was held in Fragrance Hill, a beautiful suburb of Beijing, China, during August 17–19, 2018.

In S+SSPR 2018, 49 papers contributed by authors from a multitude of different countries were accepted and presented. There were 30 oral presentations and 19 poster presentations. Each submission was reviewed by at least two and usually three Program Committee members. The accepted papers cover the major topics of current interest in pattern recognition, including classification, clustering, dissimilarity representations, structural matching, graph-theoretic methods, shape analysis, deep learning, and multimedia analysis and understanding. Authors of selected papers were invited to submit an extended version to a Special Issue on “Recent Advances in Statistical, Structural and Syntactical Pattern Recognition,” to be published in *Pattern Recognition Letters* in 2019.

We were delighted to have three prominent keynote speakers: Prof. Edwin Hancock from the University of York, who was the IAPR TC1 Pierre Devijver Award winner in 2018, Prof. Josef Kittler from the University of Surrey, and Prof. Xilin Chen from the University of the Chinese Academy of Sciences.

The workshops (S+SSPR 2018) were hosted by the School of Computer Science and Engineering, Beihang University. We acknowledge the generous support from Beihang University, which is one of the leading comprehensive research universities in China, covering engineering, natural sciences, humanities, and social sciences. We also wish to express our gratitude for the financial support provided by the Beijing Advanced Innovation Center for Big Data and Brain Computing (BDIBC), also based in Beihang University.

Finally, we would like to thank all the Program Committee members for their help in the review process. We also wish to thank all the local organizers. Without their contributions, S+SSPR 2018 would not have been successful. Finally, we express our appreciation to Springer for publishing this volume. More information about the workshops and organization can be found on the website: <http://ssspr2018.buaa.edu.cn/>.

August 2018

Xiao Bai  
Edwin Hancock  
Tin Kam Ho  
Richard Wilson  
Battista Biggio  
Antonio Robles-Kelly

# Organization

## Program Committee

Gady Agam	Illinois Institute of Technology, USA
Ethem Alpaydin	Bogazici University, Turkey
Lu Bai	University of York, UK
Xiao Bai	Beihang University, China
Silvia Biasotti	CNR - IMATI, Italy
Manuele Bicego	University of Verona, Italy
Battista Biggio	University of Cagliari, Italy
Luc Brun	GREYC, France
Umberto Castellani	University of Verona, Italy
Veronika Cheplygina	Eindhoven University of Technology, The Netherlands
Francesc J. Ferri	University of Valencia, Spain
Pasi Fränti	University of Eastern Finland, Finland
Giorgio Fumera	University of Cagliari, Italy
Michal Haindl	Institute of Information Theory and Automation of the CAS, China
Edwin Hancock	University of York, UK
Laurent Heutte	Université de Rouen, France
Tin Kam Ho	IBM Watson, USA
Atsushi Imiya	IMIT Chiba University, Japan
Jose M. Iñesta	Universidad de Alicante, Spain
Francois Jacquenet	Laboratoire Hubert Curien, France
Xiuping Jia	The University of New South Wales, Australian Defence Force Academy, Australia
Xiaoyi Jiang	University of Münster, Germany
Tomi Kinnunen	University of Eastern Finland, Finland
Jesse Krijthe	Leiden University, The Netherlands
Adam Krzyzak	Concordia University, Canada
Mineichi Kudo	Hokkaido University, Japan
Arjan Kuijper	TU Darmstadt, Germany
James Kwok	The Hong Kong University of Science and Technology, SAR China
Xuelong Li	Chinese Academy of Sciences, China
Xianglong Liu	Beihang University, China
Marco Loog	Delft University of Technology, The Netherlands
Bin Luo	Anhui University, China
Mauricio Orozco-Alzate	Universidad Nacional de Colombia, Colombia
Nikunj Oza	NASA, USA
Tapio Pahikkala	University of Turku, Finland

Marcello Pelillo	University of Venice, Italy
Filiberto Pla	Jaume I University, Spain
Marcos Quiles	Federal University of Sao Paulo, Brazil
Peng Ren	China University of Petroleum, China
Eraldo Ribeiro	Florida Institute of Technology, USA
Antonio Robles-Kelly	CSIRO, Australia
Jairo Rocha	University of the Balearic Islands, Spain
Luca Rossi	Aston University, UK
Samuel Rota Bulò	Fondazione Bruno Kessler, Italy
Punam Kumar Saha	University of Iowa, USA
Carlo Sansone	University of Naples Federico II, Italy
Frank-Michael Schleif	University of Bielefeld, Germany
Francesc Serratos	Universitat Rovira i Virgili, Spain
Ali Shokoufandeh	Drexel University, USA
Humberto Sossa	CIC-IPN, Mexico
Salvatore Tabbone	Université de Lorraine, France
Kar-Ann Toh	Yonsei University, South Korea
Ventzeslav Valev	Institute of Mathematics and Informatics Bulgarian Academy of Sciences, Bulgaria
Mario Vento	Università degli Studi di Salerno, Italy
Wenwu Wang	University of Surrey, UK
Richard Wilson	University of York, UK
Terry Windeatt	University of Surrey, UK
Jing-Hao Xue	University College London, UK
De-Chuan Zhan	Nanjing University, China
Lichi Zhang	Shanghai Jiao Tong University, China
Zhihong Zhang	Xiamen University, China
Jun Zhou	Griffith University, Australia

# Contents

## Classification and Clustering

Image Annotation Using a Semantic Hierarchy . . . . .	3
<i>Abdessalem Bouzaïeni and Salvatore Tabbone</i>	
Malignant Brain Tumor Classification Using the Random Forest Method . . . .	14
<i>Lichi Zhang, Han Zhang, Islem Rekik, Yaozong Gao, Qian Wang, and Dinggang Shen</i>	
Rotationally Invariant Bark Recognition . . . . .	22
<i>Václav Remeš and Michal Haindl</i>	
Dynamic Voting in Multi-view Learning for Radiomics Applications . . . . .	32
<i>Hongliu Cao, Simon Bernard, Laurent Heutte, and Robert Sabourin</i>	
Iterative Deep Subspace Clustering . . . . .	42
<i>Lei Zhou, Shuai Wang, Xiao Bai, Jun Zhou, and Edwin Hancock</i>	
A Scalable Spectral Clustering Algorithm Based on Landmark-Embedding and Cosine Similarity . . . . .	52
<i>Guangliang Chen</i>	

## Deep Learning and Neural Networks

On Fast Sample Preselection for Speeding up Convolutional Neural Network Training . . . . .	65
<i>Frédéric Rayar and Seiichi Uchida</i>	
UAV First View Landmark Localization via Deep Reinforcement Learning . . .	76
<i>Xinran Wang, Peng Ren, Leijian Yu, Lirong Han, and Xiaogang Deng</i>	
Context Free Band Reduction Using a Convolutional Neural Network . . . . .	86
<i>Ran Wei, Antonio Robles-Kelly, and José Álvarez</i>	
Local Patterns and Supergraph for Chemical Graph Classification with Convolutional Networks . . . . .	97
<i>Évariste Daller, Sébastien Bougleux, Luc Brun, and Olivier Lézoray</i>	
Learning Deep Embeddings via Margin-Based Discriminate Loss . . . . .	107
<i>Peng Sun, Wenzhong Tang, and Xiao Bai</i>	

**Dissimilarity Representations and Gaussian Processes**

Protein Remote Homology Detection Using Dissimilarity-Based Multiple Instance Learning. . . . . 119  
*Antonelli Mensi, Manuele Bicego, Pietro Lovato, Marco Loog, and David M. J. Tax*

Local Binary Patterns Based on Subspace Representation of Image Patch for Face Recognition. . . . . 130  
*Xin Zong*

An Image-Based Representation for Graph Classification . . . . . 140  
*Frédéric Rayar and Seiichi Uchida*

Visual Tracking via Patch-Based Absorbing Markov Chain . . . . . 150  
*Ziwei Xiong, Nan Zhao, Chenglong Li, and Jin Tang*

Gradient Descent for Gaussian Processes Variance Reduction. . . . . 160  
*Lorenzo Bottarelli and Marco Loog*

**Semi and Fully Supervised Learning Methods**

Sparsification of Indefinite Learning Models. . . . . 173  
*Frank-Michael Schleich, Christoph Raab, and Peter Tino*

Semi-supervised Clustering Framework Based on Active Learning for Real Data . . . . . 184  
*Ryosuke Odate, Hiroshi Shinjo, Yasufumi Suzuki, and Masahiro Motobayashi*

Supervised Classification Using Feature Space Partitioning. . . . . 194  
*Ventzeslav Valev, Nicola Yanev, Adam Krzyżak, and Karima Ben Suliman*

Deep Homography Estimation with Pairwise Invertibility Constraint . . . . . 204  
*Xiang Wang, Chen Wang, Xiao Bai, Yun Liu, and Jun Zhou*

**Spatio-temporal Pattern Recognition and Shape Analysis**

Graph Time Series Analysis Using Transfer Entropy . . . . . 217  
*Ibrahim Caglar and Edwin R. Hancock*

Analyzing Time Series from Chinese Financial Market Using a Linear-Time Graph Kernel . . . . . 227  
*Yuhang Jiao, Lixin Cui, Lu Bai, and Yue Wang*



A Preliminary Survey of Analyzing Dynamic Time-Varying Financial Networks Using Graph Kernels. . . . . 237  
*Lixin Cui, Lu Bai, Luca Rossi, Zhihong Zhang, Yuhang Jiao, and Edwin R. Hancock*

Few-Example Affine Invariant Ear Detection in the Wild. . . . . 248  
*Jianming Liu, Yongsheng Gao, and Yue Li*

Line Voronoi Diagrams Using Elliptical Distances . . . . . 258  
*Aysylu Gabdulkhakova, Maximilian Langer, Bernhard W. Langer, and Walter G. Kropatsch*

**Structural Matching**

Modelling the Generalised Median Correspondence Through an Edit Distance . . . . . 271  
*Carlos Francisco Moreno-García and Francesc Serratosa*

Learning the Sub-optimal Graph Edit Distance Edit Costs Based on an Embedded Model. . . . . 282  
*Pep Santacruz and Francesc Serratosa*

Ring Based Approximation of Graph Edit Distance. . . . . 293  
*David B. Blumenthal, Sébastien Bougleux, Johann Gamper, and Luc Brun*

Graph Edit Distance in the Exact Context . . . . . 304  
*Mostafa Darwiche, Romain Raveaux, Donatello Conte, and Vincent T'Kindt*

The VF3-Light Subgraph Isomorphism Algorithm: When Doing Less Is More Effective . . . . . 315  
*Vincenzo Carletti, Pasquale Foggia, Antonio Greco, Alessia Saggese, and Mario Vento*

A Deep Neural Network Architecture to Estimate Node Assignment Costs for the Graph Edit Distance . . . . . 326  
*Xavier Cortés, Donatello Conte, Hubert Cardot, and Francesc Serratosa*

Error-Tolerant Geometric Graph Similarity . . . . . 337  
*Shri Prakash Dwivedi and Ravi Shankar Singh*

Learning Cost Functions for Graph Matching . . . . . 345  
*Rafael de O. Werneck, Romain Raveaux, Salvatore Tabbone, and Ricardo da S. Torres*

**Multimedia Analysis and Understanding**

Matrix Regression-Based Classification for Face Recognition . . . . . 357  
*Jian-Xun Mi, Quanwei Zhu, and Zhiheng Luo*

Plenoptic Imaging for Seeing Through Turbulence . . . . . 367  
*Richard C. Wilson and Edwin R. Hancock*

Weighted Local Mutual Information for 2D-3D Registration  
in Vascular Interventions . . . . . 376  
*Cai Meng, Qi Wang, Shaoya Guan, and Yi Xie*

Cross-Model Retrieval with Reconstruct Hashing . . . . . 386  
*Yun Liu, Cheng Yan, Xiao Bai, and Jun Zhou*

Deep Supervised Hashing with Information Loss . . . . . 395  
*Xueni Zhang, Lei Zhou, Xiao Bai, and Edwin Hancock*

Single Image Super Resolution via Neighbor Reconstruction . . . . . 406  
*Zhihong Zhang, Zhuobin Xu, Zhiling Ye, Yiqun Hu, Lixin Cui,  
and Lu Bai*

An Efficient Method for Boundary Detection from Hyperspectral Imagery . . . 416  
*Suhad Lateef Al-Khafaji, Jun Zhou, and Alan Wee-Chung Liew*

**Graph-Theoretic Methods**

Bags of Graphs for Human Action Recognition . . . . . 429  
*Xavier Cortés, Donatello Conte, and Hubert Cardot*

Categorization of RNA Molecules Using Graph Methods. . . . . 439  
*Richard C. Wilson and Enes Algul*

Quantum Edge Entropy for Alzheimer’s Disease Analysis . . . . . 449  
*Jianjia Wang, Richard C. Wilson, and Edwin R. Hancock*

Approximating GED Using a Stochastic Generator and Multistart IPFP . . . . . 460  
*Nicolas Boria, Sébastien Bogleux, and Luc Brun*

Offline Signature Verification by Combining Graph Edit Distance  
and Triplet Networks . . . . . 470  
*Paul Maergner, Vinaychandran Pondenkandath, Michele Alberti,  
Marcus Liwicki, Kaspar Riesen, Rolf Ingold, and Andreas Fischer*

On Association Graph Techniques for Hypergraph Matching . . . . . 481  
*Giulia Sandi, Sebastiano Vascon, and Marcello Pelillo*

Directed Network Analysis Using Transfer Entropy Component Analysis. . . . 491  
*Meihong Wu, Yangbin Zeng, Zhihong Zhang, Haiyun Hong, Zhuobin Xu, Lixin Cui, Lu Bai, and Edwin R. Hancock*

A Mixed Entropy Local-Global Reproducing Kernel for Attributed Graphs. . . . 501  
*Lixin Cui, Lu Bai, Luca Rossi, Zhihong Zhang, Lixiang Xu, and Edwin R. Hancock*

Dirichlet Densifiers: Beyond Constraining the Spectral Gap . . . . . 512  
*Manuel Curado, Francisco Escolano, Miguel Angel Lozano, and Edwin R. Hancock*

**Author Index** . . . . . 523

# **Classification and Clustering**



# Image Annotation Using a Semantic Hierarchy

Abdessalem Bouzaïeni and Salvatore Tabbone<sup>(✉)</sup>

Université de Lorraine-LORIA, UMR 7503, Vandoeuvre-les-Nancy, France  
{[abdessalem.bouzaïeni](mailto:abdessalem.bouzaïeni@loria.fr), [tabbone](mailto:tabbone@loria.fr)}@loria.fr

**Abstract.** With the fast development of smartphones and social media image sharing, automatic image annotation has become a research area of great interest. It enables indexing, extracting and searching in large collections of images in an easier and faster way. In this paper, we propose a model for the annotation extension of images using a semantic hierarchy. This latter is built from vocabulary keyword annotations combining a mixture of Bernoulli distributions with mixtures of Gaussians.

**Keywords:** Graphical models · Automatic image annotation  
Multimedia retrieval · Classification

## 1 Introduction

Image annotation has been widely studied in recent years, and many approaches have been proposed [35]. These approaches can be grouped into generative models or discriminative models [13]. Generative models build a joint distribution between visual and textual characteristics of an image in order to find correspondences between image descriptors and annotation keywords. Discriminative models enable converting the problem of annotation into classification problem. Several classifiers were used for annotation such as SVM, KNN and decision trees. Most of these automatic image annotation approaches are based on the formulation of a correspondence function between low level features and semantic concepts using machine learning techniques. However, the only use of learning algorithms seems to be insufficient to surmount the semantic gap problem [11, 31], and thus to produce efficient systems for automatic image annotation. Indeed, in most image annotation approaches, the semantic is limited to its perceptual manifestation through the learning of a matching function associating low-level features with visual concepts of higher semantic level. The performances of these approaches depend on concepts number and the nature of targeted data. Thus, the use of structured knowledge, such as semantic hierarchies and ontologies, seems to be a good compromise to improve these approaches. Recently, several works have focused on the use of semantic hierarchies to annotate images [32]. These structures can be classified, as mentioned in [31], into three main categories: textual, visual and visuo-textual hierarchies. *Textual hierarchies* are

conceptual hierarchies constructed using a measure of similarity between concepts. Several approaches are based on WordNet [23] for the construction of textual hierarchies [17, 21]. Marszałek et al. [21] have proposed a hierarchy constructed by extracting the relevant subgraphs from WordNet and connecting all the concepts of the annotation vocabulary. Although approaches in this category exploit a knowledge representation to provide a richer annotation, they ignore the visual information which is very important in image annotation task. *Visual hierarchies* use low-level visual features where similar images are usually represented in the nodes and vocabulary words are represented in the leafs of the hierarchy. Bart et al. [3] have proposed a Bayesian method to find a taxonomy such that an image is generated from a path in the tree. Similar images have many common nodes on their associated paths and therefore a short distance to each other. Griffin et al. [12] built a hierarchy for a faster classification. They classified at first images to estimate a confusion matrix. Then, they grouped confounding categories in an ascending way. They also built a descendant hierarchy for the comparison by successively dividing categories. Both hierarchies showed similar results for speed and accuracy of classification. Hierarchies in this category can be used for hierarchical image classification in order to accelerate and improve classification. However, they present a major problem which is the difficulty of semantic interpretation since they are based on visual characteristics only. Textual and visual hierarchies have solved several problems by grouping objects into organized structures. They can increase the accuracy and reduce the complexity of systems [31] but they are not adequate for image annotation. Indeed, textual semantic is not always consistent with visual images, and is therefore insufficient to build good semantic structures to annotate images [34]. Visual semantics alone can not lead to a significant semantic hierarchy since it is difficult to interpret semantically. Therefore it is interesting to use these two information together to obtain semantic hierarchies well suited to image annotation task. Bannour et al. [1] have proposed a new approach for automatic construction of semantic hierarchies adapted to images classification and annotation. This method is based on the use of a similarity measure that integrates visual, conceptual and contextual information. In the same vein, Qian et al. [29] focused on annotating images in two levels by integrating both global and local visual characteristics with semantic hierarchies.

We propose in this paper a semi-automatic method of building a semantic taxonomy from the keywords of a given annotation vocabulary. This taxonomy based on the use of visual, semantic and contextual information is integrated in a probabilistic graphical model for the automatic extension of image annotation. The use of taxonomy can increase annotation performance and enrich the vocabulary used.

## 2 Building Taxonomy

A taxonomy is a collection of vocabulary terms organized into a hierarchical structure. Each term in a taxonomy is in one or more parent-child relationships

with other terms in the taxonomy. Recently, many works have been devoted to the automatic creation of a domain-specific ontology or taxonomy [10, 18]. The construction of manual taxonomy is a laborious process, and the resulting taxonomy is often subjective, compared with constructed taxonomies by data-driven approaches. In addition, automatic approaches have the potential to allow humans or even machines to understand a highly targeted and potentially scalable domain. However, the problem of taxonomy induction from a keyword set is a major challenge [18]. Although the use of a keyword set allows to more precisely characterize a specific domain, the keyword set does not contain explicit relationships from which a taxonomy can be constructed. One way to overcome this problem is to enrich the annotation vocabulary by adding new keywords. Liu et al. [18] presented a new approach which can automatically derive a domain-dependent taxonomy from a keyword set by exploiting both a general knowledge base and a keyword search. To enrich the vocabulary, they used the conceptualization technique by extracting contextual information from a search engine. The taxonomy is then constructed by hierarchical classification of the keywords using Bayesian rose tree algorithm [4]. In the rest of this section, we will present the three types of information used as well as our method of building a taxonomy from a keywords set.

## 2.1 Semantic information

Semantic information reflects the semantic significance of a given keyword from a linguistic point of view. Many machine learning algorithms are unable to process the text in its raw form. They need numbers as input to do any type of work, be it classification, regression, . . . . Intuitively, the aim is to find a vectorial representation which characterizes the linguistic significance of a given keyword. These methods usually attempt to represent a dictionary word by a real number vector. Several strategies have been proposed for word embedding but they proved to be limited in their representations until Mitolov et al. [22] introduced word2vec into the natural language processing community. Word2vec is a group of related models used to produce word embedding. These models are neural networks with two layers formed to reconstruct the linguistic contexts of the words. This model takes as input a large corpus of text and produces a vector space, typically of several hundreds of dimensions, with for each single word of the corpus a corresponding vector in space. Word vectors are positioned in the vector space so that words which share common contexts in the corpus are located near each other in the space. The Word2vec model and its applications have recently attracted a lot of attention in the machine learning community. These dense vector representations of words learned by word2vec have semantic meanings and are useful in a wide range of use cases.

## 2.2 Visual information

Visual information reflects visual appearance of a given keyword in the learning images annotated by this keyword. It is therefore a question of finding a vector

representation which makes it possible to characterize this appearance in the learning images. For a given keyword  $Kw_i$ , a set of images  $R_{Kw_i}$  is selected from the learning set  $T$  of size  $n$ . All images in the  $R$  set must be annotated by  $Kw_i$ . Thus,  $R_{Kw_i} = \bigcup_{1 \leq j \leq n} \{I_j\} / Kw_i \in W_{I_j}$ .  $W_{I_j}$  represents the set of keywords annotating the image  $I_j$  in  $T$ . For each image in the set  $R_{Kw_i}$ , interest points are detected using the SIFT detectors [19]. For each point found, a SIFT descriptor is calculated. The images are matched by minimizing the distance between their descriptors and the result of this matching is taken as visual information representing the keyword  $Kw_i$ . Thus, the visual information of a keyword  $Kw_i$ , denoted by  $Vis(Kw_i)$ , is defined by the following set:  $Vis(Kw_i) = \bigcap matching(I_i, I_j) \forall I_i, I_j \in R_{Kw_i}$ .

### 2.3 Contextuel information

Since real-world objects tend to exist in context, incorporating contextual information is important to help understand the semantics of the image. Contextual information is used to determine the context in which keywords appear by linking those that often appear together in image annotation even if they are distant visually or semantically. For example, the two keywords “horse” and “grass” can annotate together an image to represent a natural scene, while they have no visual similarity or semantic similarity since “horse” belongs to the family of animals and “Grass” belongs to the family of plants. A simple method for representing contextual information is to find the frequency of co-occurrence of a pair of keywords. This information depends only on the annotation vocabulary keywords used. Therefore, we use the mutual information to characterize the contextual information between each keyword and the whole vocabulary. This metric was used in [1]. Let  $Kw_i$  and  $Kw_j$  be two keywords. The contextual information of  $Kw_i$  and  $Kw_j$ , denoted by  $cont(Kw_i, Kw_j)$ , is defined by:  $cont(Kw_i, Kw_j) = \log \frac{P(Kw_i, Kw_j)}{P(Kw_i)P(Kw_j)}$ .  $P(Kw_i)$  represents the appearance probability of the keyword  $Kw_i$  in the database image.  $P(Kw_i, Kw_j)$  represents the joint appearance probability of the two keywords  $Kw_i$  and  $Kw_j$  together.

### 2.4 Proposed method

Once we have estimated the visual, contextual and semantic information for each vocabulary keyword, it is important to group them into a semantic taxonomy. The three type of information are used together in a single feature vector for the taxonomy construction. The taxonomy construction process is divided into three main stages: (1) *Characterization*: calculate the semantic, visual and contextual information defined in the Sects. 2.1, 2.2 and 2.3 for each keyword in vocabulary. A vector which characterizes each keyword is defined by concatenating the three types of information; (2) *Clustering*: group the closest keywords according to the information defined in a semantic group. We used K-means clustering (Euclidean distance) algorithm with normalized (using the mean and standard deviation) characteristic vectors of the keywords to group them into K groups;



(3) *Construction*: build in a bottom up manner a hierarchy for each semantic group found in the previous step. First, a new keyword is added for each of the  $K$  groups. This new keyword represents the concept or family shared by all keywords in the group. Then, arcs are added between all keywords of the group and the new added keyword. These arcs represent the parent-child relationship between the group’s keywords (children) and the newly added keyword (parent).

### 3 Annotation Model Using Taxonomy

Once the taxonomy is built, it is integrated in the probabilistic graphical model whose structure is represented in the Fig. 1. This model is a mixture of Bernoulli distributions and Gaussian mixtures. The visual characteristics of a given image are considered as continuous variables which follow a law whose density function is a Gaussian mixture density. They are modeled by two nodes: (1) The *Gaussian* node is modeled by a continuous random variable which is used to represent the computed descriptors on the image; (2) The *Component* node is modeled by a hidden random variable which is used to represent the weights of the Gaussians. It may take  $g$  different values corresponding to the number of Gaussians used in the mixture. The textual characteristics of a given image are modeled by the constructed taxonomy nodes. Each node is represented by a discrete random variable which follows a Bernoulli distribution. This variable takes two possible values: 0 and 1. The value 1 taken by the variable representing the node  $kw_i$  indicates that the image is annotated by the keyword  $i$  in the vocabulary  $New\_V$  and the value 0 indicates absence of this keyword in the image annotation. A *Class* root node is used to represent the class of image. It may take  $k$  values corresponding to the predefined classes  $C_1, \dots, C_k$ . To learn the parameters of our model, we use the EM algorithm [7]. This algorithm is the most used in the case of missing data. Given a new image  $Im_i$  represented by its visual characteristics  $VC_1, \dots, VC_M$  and its existing keywords  $Kw_1, \dots, Kw_n$ , we can use the junction tree algorithm [16] to extend the annotation of this image with other keywords. We can calculate the posterior probability:  $P(Kw_i|I_i) = P(Kw_i|VC_1, \dots, VC_M, Kw_1, \dots, Kw_n)$  and also the posterior probability:  $P(C_i|I_i) = P(C_i|VC_1, \dots, VC_M, Kw_1, \dots, Kw_n)$  to identify the class of image. The query image is assigned to the class  $C_i$  maximizing this probability. Most automatic image annotation methods assume a fixed annotation length  $k$  (usually 5) for each image. However, the fixed-length annotation may give insufficient or very long annotations. With a short length, it is possible that some content in the image will not be captured by the annotation. Unlike with a long length, it is possible that annotations generated contain words which are irrelevant to the content. Thus, to solve this problem, we can define a threshold  $\lambda$  on the probability of a keyword and an image will be annotated by a  $Kw_i$  keyword if and only if:  $P(Kw_i|I_i) > \lambda$ .

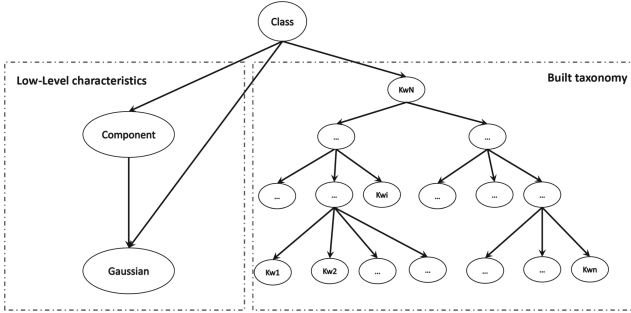


Fig. 1. Annotation model using the taxonomy.

## 4 Experimentation

In this section we present the evaluation of our model before and after the semantic hierarchy integration. We test our approach on Corel-5K dataset which is used as a benchmark in the literature for images annotation and retrieval. This dataset is divided into 4500 images for learning and 500 images for tests with a vocabulary of 260 keywords. For semantic information, we used the pre-trained Word2vec model on Google News Corpus<sup>1</sup>. The length of each vector obtained by this model is 300 characteristics. To compute the visual information of a keyword  $Kw_i$ , we need to define the set of images  $R_{Kw_i}$  from the learning dataset. Therefore, to ensure a robust visual description, we select images annotated by the smallest set of keywords (including  $Kw_i$ ) and we limit the number of images (set experimentally to 6). For the visual characteristics of each image, we used the descriptors: RGB color histogram [30], LBP [27], GIST [28] and SIFT [19]. Using visual, contextual and semantic information, we have grouped the 260 annotation vocabulary keywords of the Corel-5k database into 30 classes following the main steps defined in Sect. 2.4 and to keep a good compromise between the depth of the hierarchy and the model complexity. For each group, a new keyword is added as the parent of the group members. The parent must describe the semantic concept shared by the whole group. Thus, 30 new keywords obtained from the clustering were in turn grouped into 7 new groups. Starting with a vocabulary of 260 keywords, we obtained a new vocabulary of

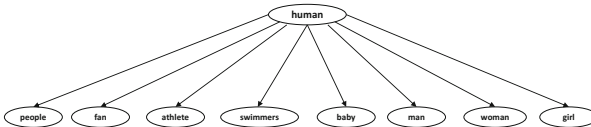


Fig. 2. Graphic representation of “human” group.

<sup>1</sup> <https://s3.amazonaws.com/dl4j-distribution/GoogleNews-vectors-negative300.bin.gz>.







**Table 1.** Performance of our model against different image annotation methods on Corel-5k dataset.

Method	Corel-5K			
	<i>P</i>	<i>R</i>	<i>F1</i>	<i>N+</i>
MBRM [9]	24	25	25	122
SVM-DMBRM [24]	36	48	41	197
NMF-KNN [15]	38	56	45	150
2PKNN [33]	44	46	45	191
CNN-R [25]	32	41	37	166
HHD [26]	31	49	38	194
MLDL [14]	45	49	47	198
SLED [5]	35	51	42	196
RFC-PSO [8]	26	22	24	109
Fuzzy [20]	27	32	29	–
Corr-LDA [6]	21	36	27	131
GMM-Mult [2]	27	38	32	154
Our method without SH	34	45	39	175
Our method with SH	42	47	44	182

298 keywords organized in a taxonomy form. This taxonomy which represents the semantic relations between keywords is added to our model as shown in Fig. 1. An example of clustering where the semantic concept “human” (added manually) shared by members of a group is shown in Fig. 2. Table 1 shows the performance of different image annotation methods on the Corel-5k database. The rows in this table are grouped according to the models used by these methods. The first group contains methods based on relevance models. The second row is focused on methods using algorithms based on nearest neighbors. The third group represents methods using deep representations based on CNN. The next row shows the performance of some methods based on sparse coding. Variety of approaches such as random forests belong to the fifth row. The last group shows the performances of methods close to our model and using probabilistic graphical models. The last two lines show the results of our method without semantic hierarchy (without SH) and with semantic hierarchy (with SH). In this table, we automatically annotated each image in the test database by 5 keywords and we calculated recall (*R*), precision (*P*), *F1* and *N+* measures. Our method provides competitive results compared to state-of-the-art methods.

Indeed, it surpasses all the methods of the first and fifth group. It also gives good results compared to the methods of the second group which use KNN. However, these methods have the disadvantage of a large annotation time. Indeed, each image to be annotated must be compared to all the images of the database. On the contrary, for our method, the learning is done once at all, and to annotate an image, we calculate the posterior probabilities only (see Sect. 3). In addition, these methods suffer from the problem of choosing the number of neighbors and the distance to use between visual characteristics. Although third group methods

using deep learning offer good performance and reduce low-level feature calculations, these algorithms require a large amount of data in the learning phase and require more computing power and storage. Compared to the methods listed in the Table 1, except for the last group, our method has the advantage to be used for the two tasks of image annotation and classification. Another advantage of our model is the interpretation of the network structure which provides valuable information about conditional dependence between variables. We observe that the performances of our model are better than those close to our approach. The superiority compared to Corr-LDA [6] is justified by the fact that we use a mixture of multivariate Gaussians whereas this model uses a multivariate Gaussian. Moreover, the addition of semantic relationships between keywords and the use of more relevant visual characteristics increase the performance of our approach compared to GMM-Mult [2]. We also note that the integration of the semantic hierarchy into the model considerably increases the performance of annotations and especially in terms of precision. Indeed, we obtained a precision of 34% with the old model (“Our method without SH” in the table) and after the integration of the semantic hierarchy, we reach a precision of 42% (“Our method with SH” in the table). Another advantage of our approach is the possibility to enrich the annotation by using new keywords which did not belong to the initial annotation vocabulary, unlike the fourth group method in the Table 1. Figure 3 illustrates the annotation of some images of Corel-5k database where labels of the ground truth are given. We notice that the images are not annotated by the same number

 <p>sky, sun, clouds, tree sky, sun, clouds, tree, palm, natural_view, shaft, natural_phenomenon, nature</p>	 <p>sky, jet, plane sky, jet, plane, f-16, aviation, natural_view, transport, nature</p>	 <p>bear, polar, snow, tundra bear, polar, snow, ice, various_animal, extreme_environment, animal</p>
 <p>water, boats, bridge water, boats, bridge, arch, pyramid, natural_resource, town, structure, architectures, nature</p>	 <p>tree, horses, mare, foals tree, horses, mare, foals, field, herbivorous_animal, shaft, animal, nature</p>	 <p>sky, buildings, flag sky, buildings, skyline, architectural_element, natural_view, architectures, street, nature</p>

**Fig. 3.** Examples of image annotation using the semantic hierarchy for Corel-5k.

of keywords because of the use of threshold  $\lambda$  experimentally defined at 0.75. We also notice that new keywords appear which do not belong to the initial vocabulary. For example, the fourth image is annotated manually by three keywords (“water”, “boats” and “bridge”), seven new keywords (“arch”, . . . and “nature”) are automatically added after the automatic annotation extension. The two keywords (“arch” and “pyramid”) belong to the initial annotation vocabulary and the other five keywords belong to the new added vocabulary.

## 5 Conclusion

In this paper, we presented a semi-automatic method for building a semantic hierarchy from a set of keywords. This hierarchy is based on the use of visual, contextual and semantic information for each keyword. After building the hierarchy, we integrated it into a probabilistic graphical model decomposed into a mixture of Bernoulli distributions and Gaussian mixtures. The integration of the constructed semantic hierarchy in the model greatly increases the performance of annotations. The obtained results are competitive compared to state-of-the-art methods. In addition, we can enrich the image annotation by using new keywords which did not belong to the initial annotation vocabulary. In future works, we want to automate the semantic hierarchy construction where new concepts could be added automatically.

## References

1. Bannour, H., Hudelot, C.: Building and using fuzzy multimedia ontologies for semantic image annotation. *Multimed. Tools Appl.* **72**, 2107–2141 (2014)
2. Barrat, S., Tabbone, S.: Classification and automatic annotation extension of images using Bayesian network. In: da Vitoria Lobo, N., et al. (eds.) *SSPR/SPR 2008*. LNCS, vol. 5342, pp. 937–946. Springer, Heidelberg (2008). [https://doi.org/10.1007/978-3-540-89689-0\\_97](https://doi.org/10.1007/978-3-540-89689-0_97)
3. Bart, E., Porteous, I., Perona, P., Welling, M.: Unsupervised learning of visual taxonomies. In: *CVPR*, pp. 1–8. IEEE (2008)
4. Blundell, C., Teh, Y.W., Heller, K.A.: Bayesian rose trees. *arXiv preprint arXiv:1203.3468* (2012)
5. Cao, X., Zhang, H., Guo, X., Liu, S., Meng, D.: SLED: semantic label embedding dictionary representation for multilabel image annotation. *IEEE IP* **24**(9), 2746–2759 (2015)
6. Chong, W., Blei, D., Li, F.F.: Simultaneous image classification and annotation. In: *CVPR*, pp. 1903–1910. IEEE (2009)
7. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. *JRSS Ser. B* **39**(1), 1–38 (1977)
8. El-Bendary, N., Kim, T.H., Hassanien, A.E., Sami, M.: Automatic image annotation approach based on optimization of classes scores. *Computing* **96**(5), 381–402 (2014)
9. Feng, S., Manmatha, R., Lavrenko, V.: Multiple Bernoulli relevance models for image and video annotation. In: *CVPR*, vol. 2, pp. 1002–1009. IEEE (2004)

10. Fountain, T., Lapata, M.: Taxonomy induction using hierarchical random graphs. In: ACL, pp. 466–476 (2012)
11. Fu, H., Zhang, Q., Qiu, G.: Random forest for image annotation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7577, pp. 86–99. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-33783-3\\_7](https://doi.org/10.1007/978-3-642-33783-3_7)
12. Griffin, G., Perona, P.: Learning and using taxonomies for fast visual categorization. In: CVPR, pp. 1–8. IEEE (2008)
13. Ji, P., Gao, X., Hu, X.: Automatic image annotation by combining generative and discriminant models. *Neurocomputing* **236**, 48–55 (2017)
14. Jing, X.Y., Wu, F., Li, Z., Hu, R., Zhang, D.: Multi-label dictionary learning for image annotation. *IEEE Trans. Image Process.* **25**(6), 2712–2725 (2016)
15. Kalayeh, M.M., Idrees, H., Shah, M.: NMF-KNN: image annotation using weighted multi-view non-negative matrix factorization. In: CVPR, pp. 184–191 (2014)
16. Lauritzen, S.L., Spiegelhalter, D.J.: Local computations with probabilities on graphical structures and their application to expert systems. *JRSS Ser. B* **50**(2), 157–224 (1988)
17. Li, L.J., Socher, R., Fei-Fei, L.: Towards total scene understanding: classification, annotation and segmentation in an automatic framework. In: CVPR, pp. 2036–2043. IEEE (2009)
18. Liu, X., Song, Y., Liu, S., Wang, H.: Automatic taxonomy construction from keywords. In: ACM SIGKDD, pp. 1433–1441. ACM (2012)
19. Low, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision, vol. 2, pp. 1150–1157 (1999)
20. Maihami, V., Yaghmaee, F.: Fuzzy neighbor voting for automatic image annotation. *JECEI* **4**(1), 1–8 (2016)
21. Marszałek, M., Schmid, C.: Semantic hierarchies for visual object recognition. In: CVPR (2007)
22. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint [arXiv:1301.3781](https://arxiv.org/abs/1301.3781) (2013)
23. Miller, G.A.: WordNet: a lexical database for English. *Commun. ACM* **38**(11), 39–41 (1995)
24. Murthy, V.N., Can, E.F., Manmatha, R.: A hybrid model for automatic image annotation. In: ICMR, pp. 369–376. ACM (2014)
25. Murthy, V.N., Maji, S., Manmatha, R.: Automatic image annotation using deep learning representations. In: ICMR, pp. 603–606. ACM (2015)
26. Murthy, V.N., Sharma, A., Chari, V., Manmatha, R.: Image annotation using multi-scale hypergraph heat diffusion framework. In: ICMR. ACM (2016)
27. Ojala, T., Pietikäinen, M., Harwood, D.: A comparative study of texture measures with classification based on featured distributions. *PR* **29**(1), 51–59 (1996)
28. Oliva, A., Torralba, A.: Modeling the shape of the scene: a holistic representation of the spatial envelope. *Int. J. Comput. Vis.* **42**(3), 145–175 (2001)
29. Qian, Z., Zhong, P., Chen, J.: Integrating global and local visual features with semantic hierarchies for two-level image annotation. *Neurocomputing* **171**, 1167–1174 (2016)
30. Swain, M.J., Ballard, D.H.: Color indexing. *IJCV* **7**(1), 11–32 (1991)
31. Tusch, A.M., Herbin, S., Audibert, J.Y.: Semantic hierarchies for image annotation: a survey. *PR* **45**(1), 333–345 (2012)
32. Uricchio, T., Ballan, L., Seidenari, L., Bimbo, A.D.: Automatic image annotation via label transfer in the semantic space. *PR* **71**, 144–157 (2017)

33. Verma, Y., Jawahar, C.V.: Image annotation using metric learning in semantic neighbourhoods. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7574, pp. 836–849. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-33712-3\\_60](https://doi.org/10.1007/978-3-642-33712-3_60)
34. Wu, L., Hua, X.S., Yu, N., Ma, W.Y., Li, S.: Flickr distance: a relationship measure for visual concepts. TPAMI **34**(5), 863–875 (2012)
35. Zhang, D., Islam, M.M., Lu, G.: A review on automatic image annotation techniques. PR **45**(1), 346–362 (2012)



# Malignant Brain Tumor Classification Using the Random Forest Method

Lichi Zhang<sup>1</sup>, Han Zhang<sup>2</sup>, Islem Rekik<sup>3</sup>, Yaozong Gao<sup>4</sup>,  
Qian Wang<sup>1</sup>, and Dinggang Shen<sup>2</sup>(✉)

<sup>1</sup> Institute for Medical Imaging Technology, School of Biomedical Engineering,  
Shanghai Jiao Tong University, Shanghai, China

<sup>2</sup> Department of Radiology and BRIC, University of North Carolina at Chapel  
Hill, Chapel Hill, USA

dinggang\_shen@med.unc.edu

<sup>3</sup> Department of Computing, University of Dundee, Dundee, UK

<sup>4</sup> Shanghai United Imaging Intelligence Co., Ltd., Shanghai, China

**Abstract.** Brain tumor grading is pivotal in treatment planning. Contrast-enhanced T1-weighted MR image is commonly used for grading. However, the classification of different types of high-grade gliomas using T1-weighted MR images is still challenging, due to the lack of imaging biomarkers. Previous studies only focused on simple visual features, ignoring rich information provided by MR images. In this paper, we propose an automatic classification pipeline using random forest to differentiate the WHO Grade III and Grade IV gliomas, by extracting discriminative features based on 3D patches. The proposed pipeline consists of three main steps in both the training and the testing stages. *First*, we select numerous 3D patches in and around the tumor regions of the given MR images. This can suppress the intensity information from the normal region, which is trivial for the classification process. *Second*, we extract features based on both patch-wise information and subject-wise clinical information, and then we refine this step to optimize the performance of malignant tumor classification. *Third*, we incorporate the classification forest for training/testing the classifier. We validate the proposed framework on 96 malignant brain tumor patients that consist of both Grade III ( $N = 38$ ) and Grade IV gliomas ( $N = 58$ ). The experiments show that the proposed framework has demonstrated its validity in the application of high-grade gliomas classification, which may help improve the poor prognosis of high-grade gliomas.

## 1 Introduction

Brain tumor is generally caused by uncontrollable cell reproductions, which has become one of the major causes of death among people. The benign and malignant brain tumors differ on the growth speed. Specifically, the benign tumors grow much slower than the malignant tumors, and do not spread to the neighboring tissues. On the other hand, the malignant tumors are more invasive, and have high chances of spreading to adjacent regions [1] and recurring after resection.

It is highly demanded to achieve preclinical assessment of the brain tumors such as grade, location, size, and border [2]. This can greatly help neurosurgeons administer



treatments to patients. Conventional classification methods include biopsy, lumbar puncture and etc., which is both time consuming and invasive. Hence, automatic classification of the tumor based on pre-surgical images using computer-aided technologies may contribute to improving tumor prognosis. However, the main challenges of tumor classification are attributed to high variations in the tumor location, size, and complex shape. There have been numerous attempts in recent years for classifying benign and malignant tumors using statistical and machine learning techniques, such as Fisher linear discriminant analysis [3], k-nearest neighbor decision tree [4], multilayer perceptron [5], support vector machine [6], and artificial neural network [7]. Further detailed literature survey of tumor classification can be found in [8].

Currently about 45% of the brain tumors are recognized as gliomas. According to the fourth edition of World Health Organization (WHO) grading scheme, gliomas are classified into malignant tumors. Among them, high-grade gliomas are more fatal and can be further classified into two types, named as WHO Grade III (including anaplastic astrocytoma and anaplastic oligodendroglioma), and WHO Grade IV (glioblastoma multiform). Differentiating the two types of high-grade gliomas is much more challenging, as they share similar imaging properties, e.g., both of them have enhanced contrast in the most commonly used contrast-enhancement T1-weighted MR imaging. It is noted that few literature has focused on the classification of the high-grade tumors.

Our goal in this paper is to alleviate the problems in classifying high-grade gliomas using only T1-weighted MR images. We hypothesize that there are discriminative features contained in this modality, which are complex and cannot be extracted using conventional classification approaches. We therefore devise a novel framework for WHO grading classification of high-grade gliomas based on contrast-enhancement T1-weighted MR imaging. Specifically, we focus only on the intensity appearances in the tumor and its surrounding regions, instead of extracting features from the whole brain. This can optimize the obtained features and suppress the undesired noise from the rest normal regions. Also, we follow a 3D patch-based strategy to implement the classification, in order to alleviate the issues caused by the high variances of tumors' shapes and locations in different patients. State succinctly, the classifier is trained from the 3D cubic patches in the training images, which is then applied to predict the grading information of the selected patches in the testing images. All the estimated results from the patches are then combined together to obtain the final classification predictions.

It is also noted that the features employed in training/testing the classifier are not only the intensity-based features extracted from the patches (i.e., patch-wise features), but also the demographic and general clinical information of the patients (e.g., age, gender and tumor size, which are subject-wise features). Both sources of features are combined for classification, which is implemented by adopting the random forest method. The main advantage of the random forest technique is that it can handle a large number of images, and provide fast and relatively accurate classification performance. Besides, it has strong robustness to the noise information and is designed to prevent overfitting issues, which definitely fits our needs.

To fulfill the goals mentioned above, there are generally three steps in the proposed framework. *First*, numerous 3D patches are selected within and around the tumor regions of the given MR images. *Second*, the feature extraction process is implemented based on both patch-wise and subject-wise features. *Third*, the classification forest

technique is utilized for training/testing the classifier. The strategies proposed in this paper are optimized for the case of high-grade gliomas classification.

## 2 Method

In this section, we present the detailed description of the learning based framework, which consists of the training and the testing stages. In the training stage, the training images containing grading information are used to train the classifiers, while as in the testing stage the trained forest is applied to predict the grading information of the input images. Both the training and testing images follow the three steps mentioned in Sect. 1 to train/test the classifiers. The detailed descriptions of the processes are presented in the subsequent sections.

### 2.1 Patch Extraction

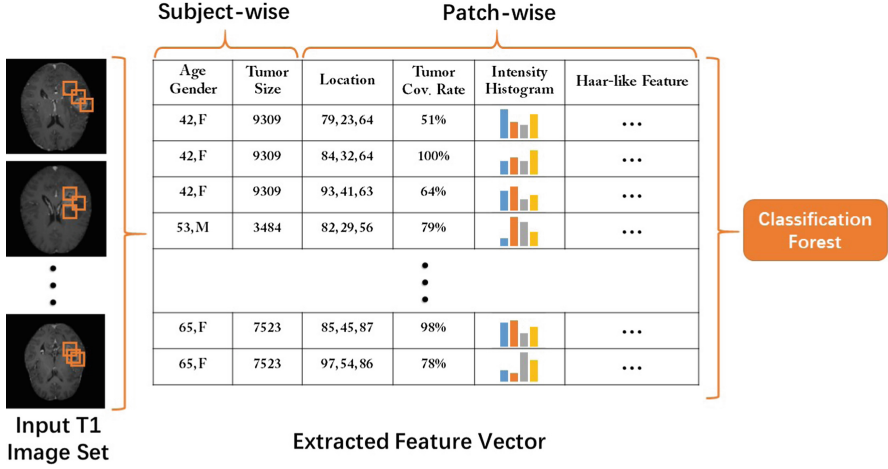
Given the set of input T1-weighted MR images with their corresponding tumor label maps, we randomly extract the group of 3D cubic patches from them. We follow the importance sampling strategy introduced in [9] to avoid the large overlapping between any pair of selected patches, since this will lead to highly-redundant information that may affect the subsequent learning process.

The strategy for the patch extraction is given as follows. First, we expand the tumor region by performing dilation process to the given label maps, and the patches are selected within the dilated area. Therefore, the information in the boundary and the surrounding area is also included for the afterward process, which may have equal importance in tumor grading classification. We also construct a probability map, which represents the priority distribution of individual voxels/patches selected for training. The probability map is initialized that the dilated tumor region is marked as 1, whilst the rest as 0. When a patch is selected, this patch region in the probability map is marked and the probability values for following patch selection is reduced. This strategy can suppress future selection of the neighboring patches, therefore preventing the overlapping issues as mentioned above. In each intensity image, we select  $m$  patches. Thus, the total number of the 3D patches in  $n$  input images is  $m \times n$ . The set of patches is denoted as  $P = \{p_1, p_2, \dots, p_{m \times n}\}$ .

### 2.2 Feature Extraction

Figure 1 illustrates the process of feature extraction after patches are obtained from the input images. Denote the  $i$ -th image  $I_i$  with its set of patches  $P^i = \{p_1^i, p_2^i, \dots, p_m^i\}$ , each patch has its corresponding feature information, which is combined together in the form of feature vector.

There are two types of features designed in this work: *subject-wise* and *patch-wise* ones. The subject-wise features are identical for all patches belonging to the same image from the same subject, which contain the general information of the corresponding patients: age, gender and tumor size. The patch-wise features, on the other hand, include the information relevant to the patch itself. There are four categories of



**Fig. 1.** The feature extraction process from the obtained patches. The feature vector consists of two types of information: *subject-wise* and *patch-wise*. The *subject-wise* features include the background information of the patients, such as age, gender and tumor size. The *patch-wise* features describe the information for the extracted patches, such as tumor cover rate, intensity histogram and Haar-like features.

data for the patch-wise features. The experiments show that they can generally represent the patch information and help in the classification processes:

- (1) **Location** of the patch center;
- (2) **Tumor coverage rate**, which shows the percentage of the patch region that is actually occupied by the tumor. This information can better describe the patches located in the boundary area;
- (3) **Intensity histogram**, representing the intensity distribution within the patch region;
- (4) **Intensity feature** of the patch, containing the details of the intensity information extracted by the Haar-like operators.

In this paper, we apply the 3D Haar-like operators to extract more complex intensity-based features due to computational efficiency and simplicity [10]. For the patch  $p$  with its region  $R$ , we randomly find two cubic areas  $R_1$  and  $R_2$  within  $R$ . The sizes of the cubic regions are randomly chosen from an arbitrary range of  $\{1, 3, 5\}$  in voxels. There are two ways to compute the Haar-like features: (1) the local mean intensity in  $R_1$ , or (2) the difference of local mean intensities in  $R_1$  and  $R_2$  [11]. The Haar-like feature operator can be thus given as [12]:

$$f_{\text{Harr}}(p) = \frac{1}{|R_1|} \sum_{u \in R_1} p(u) - \delta \frac{1}{|R_2|} \sum_{v \in R_2} p(v), \quad (1)$$

$$R_1 \subseteq R_2, R_2 \subseteq R, \delta \in \{0, 1\},$$

where  $f_{\text{Haar}}(p)$  is a Haar-like feature for the patch  $p$ , and the parameter  $\delta$  is 0 or 1 to determine the selection of one or two cubic regions.

### 2.3 Classification Forest

In this section we present detailed descriptions of the classification forest in the training and testing stages. The random forest is an ensemble of a groups of decision trees. Based on the uniform bagging strategy [13], each tree is trained using a subset of training samples with only a subset of features randomly selected from a large feature pool. Since the randomness is injected into the training process, the over-fitting problems can therefore be avoided, and also the robustness can be improved in the classification performance. Note that although the patches are randomly extracted from the images as mentioned in Sect. 2.1, to reduce computation complexity, each tree is trained using features extracted from the whole set of obtained patches.

It is also noted that the parameter values to compute the Haar-like features are randomly decided during the training stage, which are stored for future use in the testing stage. In this way, we can avoid the costly computation of the entire feature pool and then efficiently sample features from the pool.

In the training stage, each decision tree  $T_j$  learns a weak class predictor  $g(h|f(p), T_j)$  [14], where  $p$  is the input patch,  $h$  is the grading label, and  $f(p)$  the obtained feature vector combined with the 3D Haar-like features and the other features in Sect. 2.2. There are two types of nodes in the trained decision trees, which are the internal node and the leaf node. Starting with the complete set of patches  $P$  at the root (internal) node, its split function can be optimized to divide the input set into the left or right child (internal) node based on their features. The split function is developed to maximize the information gain of splitting the obtained feature vector [13]. Note that the settings of the optimal split functions are also stored in the internal node for testing. Then, the tree recursively computes the split in each of the child (internal) nodes and further divides the input patch set. It keeps growing until either reaches the maximum tree depth, or the number of training patches belonging to the internal nodes is less than a pre-defined threshold value. Then, each partition set of patches are stored in its corresponding leaf nodes  $l$  with its predictor  $g_l(h|f(p), T_j)$  computed by averaging the values of the patches [12].

In the testing stage, the strategy of patch classification is given as follows. Denote the forest that consists of  $b$  trained decision trees as  $\mathbf{F} = \{T_1, T_2, \dots, T_b\}$ , the test patch  $p_i$  for the test image  $I'$  is first pushed separately into the root nodes of each tree  $T_j$ . Guided by the learned splitting functions in the training stage, for each tree  $T_j$ , the patch will arrive at a certain leaf node, and the corresponding probability result can thus be obtained by  $g(h|f(p), T_j)$ . The overall probability from the forest  $\mathbf{F}$  can be estimated by averaging the obtained probability results from all trees, i.e.,

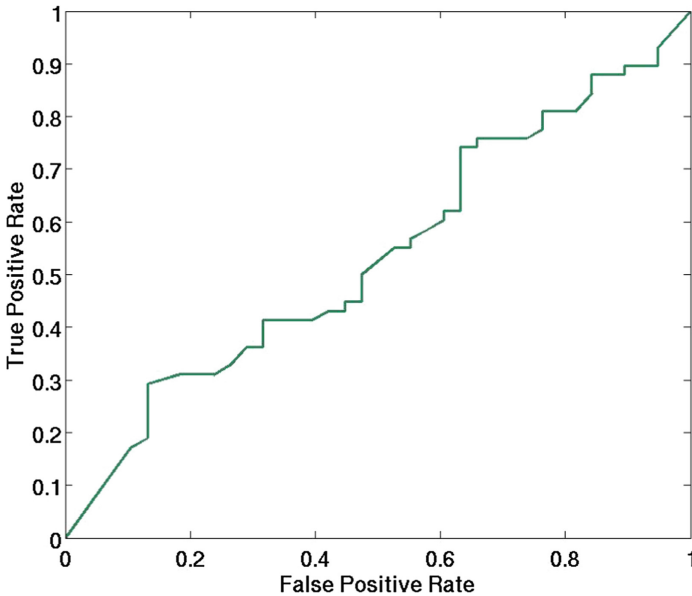
$$g(h|p_i, \mathbf{F}) = \frac{1}{b} \sum_{j=1}^b g(h|f(p_i), T_j). \quad (2)$$

The final classification estimation for the test image  $I'$  can be measured by simply averaging all probability values from all patches, which is written as:

$$g(h|I') = \sum_{i=1}^m \frac{g(h|p_i, \mathbf{F})}{m}. \quad (3)$$

### 3 Experimental Results

In this section, we evaluate the proposed framework for classifying the Grade III and Grade IV gliomas using contrast enhanced T1-weighted MR images. The dataset contains 96 MR images from patients diagnosed with high-grade gliomas intra-operatively (age  $51 \pm 15$  years, 37 males), which are acquired from a 3.0 T MR scanners. The diagnosis, i.e., tumor grading, was achieved by biopsy and histopathology. All images were pre-processed following the standard pipeline introduced in [15]. Further, we applied non-rigid registration by using SPM8<sup>1</sup> toolkit, to warp all images into the standard space. We also implemented the ITK-based histogram matching program to the acquired images, which were rescaled to a uniform intensity range [0 255]. The gliomas regions were manually segmented by experts.



**Fig. 2.** The ROC curve of the classifier.

For evaluation, we used 8-fold cross-validation setting. Basically, the 96 input MR images are randomly divided into 8 groups with equal size. In each fold, we select one fold as testing images, and the rest as training images. Also note that we follow the same parameter settings in each fold of the experiments. The parameter settings are

<sup>1</sup> <http://www.fil.ion.ucl.ac.uk/spm/software/spm8/>.

optimized by considering its fitness to the conducted experiments and the computation cost. In each image, we select 600 patches with the size of  $15 \times 15 \times 15 \text{ mm}^3$ . There are 15 trees trained in the forest, the maximum depth of each tree is set to 20, each leaf node has a minimum of eight samples, and the number of Haar features is 1000.

We provide the classification results using the evaluation metrics of sensitivity (SEN), specificity (SPE) and accuracy (ACC), which are 75.86%, 34.21% and 59.38%, respectively. Also, Fig. 2 shows the receiver operating characteristic (ROC) curve representing the performance of the trained classifier, which is created by plotting the true positive rate (TPR) against the false positive rate (FPR). It is also noted that the average runtime of the classification process is around 15 min using a standard computer (Intel Core i7-3610QM 2.30 GHz, 8 GB RAM).

## 4 Conclusion

In this paper, we present a novel framework using random forest to differentiate between WHO Grade III and Grade IV gliomas. We provide detailed descriptions of the three steps applied in both training and testing stages, which are patch extraction, feature extraction and classifier training/testing. We demonstrate experimentally that the proposed framework is capable of classifying high-grade gliomas using the commonly acquired MR images.

In the future works we intend to further explore other feature descriptors, such as local binary pattern (LBP), histogram of oriented gradients (HOG), and find out if they can be suitable to be applied in the proposed framework. We will also include the feature selection process to optimize the extracted features from the patches, which is expected to further improve the classification performance. Furthermore, we will use multimodality images (including Diffusion Tensor Imaging and resting-state functional MR Imaging) in the classification works, whose output results will be compared with those reported in this paper to assess their value for glioma grading.

## References

1. John, P.: Brain tumor classification using wavelet and texture based neural network. *Int. J. Sci. Eng. Res.* **3**, 1–7 (2012)
2. Huo, J., et al.: CADrx for GBM brain tumors: predicting treatment response from changes in diffusion-weighted MRI. *Algorithms* **2**, 1350–1367 (2009)
3. Sun, Z.-L., Zheng, C.-H., Gao, Q.-W., Zhang, J., Zhang, D.-X.: Tumor classification using eigengene-based classifier committee learning algorithm. *IEEE Sign. Process. Lett.* **19**, 455–458 (2012)
4. Wang, S.-L., Zhu, Y.-H., Jia, W., Huang, D.-S.: Robust classification method of tumor subtype by using correlation filters. *IEEE/ACM Trans. Comput. Biol. Bioinf. (TCBB)* **9**, 580–591 (2012)
5. Gholami, B., Norton, I., Eberlin, L.S., Agar, N.Y.: A statistical modeling approach for tumor-type identification in surgical neuropathology using tissue mass spectrometry imaging. *IEEE J. Biomed. Health Inf.* **17**, 734–744 (2013)

6. Sridhar, D., Murali Krishna, I.V.: Brain tumor classification using discrete cosine transform and probabilistic neural network. In: International Conference on Signal Processing Image Processing & Pattern Recognition (ICSIPR), pp. 92–96. IEEE (2013)
7. Kharat, K.D., Kulkarni, P.P., Nagori, M.: Brain tumor classification using neural network based methods. *Int. J. Comput. Sci. Inf.* **1**, 2231–5292 (2012)
8. Bauer, S., Wiest, R., Nolte, L.-P., Reyes, M.: A survey of MRI-based medical image analysis for brain tumor studies. *Phys. Med. Biol.* **58**, R97 (2013)
9. Wang, Q., Wu, G., Yap, P.-T., Shen, D.: Attribute vector guided groupwise registration. *NeuroImage* **50**, 1485–1496 (2010)
10. Viola, P., Jones, M.J.: Robust real-time face detection. *Int. J. Comput. Vis.* **57**, 137–154 (2004)
11. Han, X.: Learning-boosted label fusion for multi-atlas auto-segmentation. In: Machine Learning in Medical Imaging, pp. 17–24 (2013)
12. Wang, L., et al.: LINKS: learning-based multi-source IntegratioN framework for Segmentation of infant brain images. *NeuroImage* **108**, 160–172 (2015)
13. Breiman, L.: Random forests. *Mach. Learn.* **45**, 5–32 (2001)
14. Criminisi, A., Shotton, J., Konukoglu, E.: Decision forests: a unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Found. Trends@ Comput. Graph. Vis.* **7**, 81–227 (2012)
15. Coupé, P., Manjón, J.V., Fonov, V., Pruessner, J., Robles, M., Collins, D.L.: Patch-based segmentation using expert priors: application to hippocampus and ventricle segmentation. *NeuroImage* **54**, 940–954 (2011)



# Rotationally Invariant Bark Recognition

Václav Remeš and Michal Haindl<sup>(✉)</sup> 

The Institute of Information Theory and Automation,  
Czech Academy of Sciences, Prague, Czech Republic  
{remes,haindl}@utia.cz  
<http://www.utia.cz/>

**Abstract.** An efficient bark recognition method based on a novel wide-sense Markov spiral model textural representation is presented. Unlike the alternative bark recognition methods based on various gray-scale discriminative textural descriptions, we benefit from fully descriptive color, rotationally invariant bark texture representation. The proposed method significantly outperforms the state-of-the-art bark recognition approaches in terms of the classification accuracy.

**Keywords:** Bark recognition · Tree taxonomy classification  
Spiral Markov random field model

## 1 Introduction

Automatic bark recognition is a challenging but practical plant taxonomy application which allows fast and non-invasive tree recognition irrespective of the growing season, i.e., whether a tree has or has not its leaves, fruit, needles, or seeds or if the tree is healthy growing or just a dead stump. Automatic bark recognition makes identification or learning of tree species possible without any botanical expert knowledge through, e.g., using a dedicated mobile application. Manual identification of a tree's species based on a botanical key of bark images is a tedious task which would normally consist of scrolling through a book. Since bark can not be described as easily as leaves or needles [5, 18], the user has to go through the whole bark encyclopedia looking for the corresponding bark image.

An advantage of bark based features is their relative stability during the corresponding tree's life time. Single shrubs or trees have specific bark which can be advantageously used for their identification. It enables numerous ecological applications such as plant resource management or fast identification of invading tree species. Industrial applications can be in saw mills or bark beetle tree infestation detection.

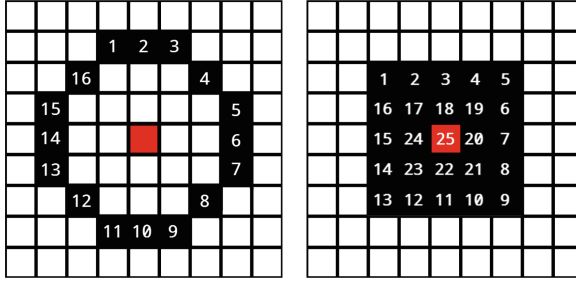
### 1.1 Alternative Bark Recognition Methods

A SVM type of classifier and gray-scale LBP features are used in [1]. Their dataset is a collection of 40 images per species and there are 23 species, i.e., a



total of 920 bark color images of local, mostly dry subtropical-climate, shrubs and trees (acacias, agaves, opuntias, palms). The classifier exploited in [9] is a radial basis probabilistic neural network. The method uses Daubechies 3rd level wavelet based features applied to each color band in the  $YC_bC_r$  color space. A similar method [8] with the same classifier uses Gabor wavelet features. Both methods use the same test set which contains 300 color bark images. Gabor banks features with a narrow-band signal model in 1-NN classifier was proposed in [4]. The test set has 8 species with 25 samples per tree category. The author also demonstrates a significant, but expectable, performance improvement when color information was added. The 1-NN and 4-NN classifier [19] represent bark textures by the run length, Haralick's co-occurrence matrix based, and histogram features. These methods are verified on a limited dataset of 160 samples from 9 species. Authors in [3] propose a rotationally invariant statistical radial binary pattern (SRBP) descriptor to characterize a bark texture. Four types of multi-scale LBP features (Multi-Block LBP (MBLBP) with a mean filter, LBP Filtering (LBPf), Multi-Scale LBP (MSLBP) with a low pass Gaussian filter, and Pyramid-based LBP (PLPB) with a pyramid transform) are used in [2]. Two bark image datasets (AFF [5], Trunk12 [17]) were used to evaluate the multi-scale LBP descriptors based bark recognition. The authors observed that multi-scale LBP provides more discriminative texture features than basic and uniform LBP and that LBPf gives the best results over all the tested descriptors on both datasets. The paper [15] proposes a combination of two types of texture features, the gray-level co-occurrence matrix metrics and the long connection length emphasis [15] binary texture features. Eighteen tree species in 90 images are classified using the k-NN classifier. The support vector machine classifier and multiscale rotationally invariant LBP features are used in [16]. The multi-class classification problem is solved using the one versus all scheme. The method is verified on two general texture datasets and the AFF bark dataset [5]. A comparison of the usefulness of the run-length method (5 features), co-occurrence correlation method (100) features for the bark k-NN classification into nine categories with 15 samples per category is presented in [19]. The method [5] uses support vector machine classifier with radial basis function kernel applied with four (contrast, correlation, homogeneity, and energy) gray-level co-occurrence matrices (GLCM), SIFT based bag-of-words, and wavelet features. The bark dataset (AFF bark dataset) consists of 1183 images of the eleven most common Austrian trees (Sect. 4). Color descriptor based on three-dimensional adaptive sum and difference histograms was applied BarTex textures in [13,14].

The majority of the published methods suffer from neglecting spectral information and using discriminative and thus approximate textural features only. Few attempts to use multispectral information [8,9,11,19] independently apply monospectral features on each spectral band or apply the color LBP features [7,12]. Most methods use private and very restricted bark databases, thus the published results are mutually incomparable and of limited information value.



**Fig. 1.** The paths of the two “spirals” in an image. Left: octagonal, right: rectangular. The numbers designate the order in which the pixels  $r$ , i.e.,  $I_r^{cs}$  neighborhoods are traversed and the red square means the center pixel. (Color figure online)

## 2 Spiral Markovian Texture Representation

The spiral adaptive 2D causal auto-regressive random (2DSCAR) field model is a generalization of the 2DCAR model [6]. The model’s functional contextual neighbour index shift set is denoted  $I_r^{cs}$ . The model can be defined in the following matrix equation:

$$Y_r = \gamma Z_r + e_r, \quad (1)$$

where  $\gamma = [a_1, \dots, a_\eta]$  is the parameter vector,  $\eta = \text{cardinality}(I_r^{cs})$ ,  $r = [r_1, r_2]$  is spatial index denoting history of movements on the lattice  $I$ ,  $e_r$  denotes driving white Gaussian noise with zero mean and a constant but unknown variance  $\sigma^2$ , and  $Z_r$  is a neighborhood support vector of  $Y_{r-s}$  where  $s \in I_r^{cs}$ .

All 2DSCAR model statistics can be efficiently estimated analytically [6]. The Bayesian parameter estimation (conditional mean value)  $\hat{\gamma}$  can be accomplished using fast, numerically robust and recursive statistics [6], given the known 2DSCAR process history  $Y^{(t-1)} = \{Y_{t-1}, Y_{t-2}, \dots, Y_1, Z_t, Z_{t-1}, \dots, Z_1\}$ :

$$\hat{\gamma}_{t-1}^T = V_{zz(t-1)}^{-1} V_{zy(t-1)}, \quad (2)$$

$$V_{t-1} = \tilde{V}_{t-1} + V_0, \quad (3)$$

$$\tilde{V}_{t-1} = \begin{pmatrix} \sum_{u=1}^{t-1} Y_u Y_u^T & \sum_{u=1}^{t-1} Y_u Z_u^T \\ \sum_{u=1}^{t-1} Z_u Y_u^T & \sum_{u=1}^{t-1} Z_u Z_u^T \end{pmatrix} = \begin{pmatrix} \tilde{V}_{yy(t-1)} & \tilde{V}_{zy(t-1)}^T \\ \tilde{V}_{zy(t-1)} & \tilde{V}_{zz(t-1)} \end{pmatrix}, \quad (4)$$

where  $t$  is the traversing order index of the sequence of multi-indices,  $r$  is based on the selected model movement in the lattice  $I$  (see Fig. 1),  $V_0$  is a positive definite initialization matrix (see [6]). The optimal causal functional contextual neighbourhood  $I_r^{cs}$  can be solved analytically by a straightforward generalisation of the Bayesian estimate in [6]. The model can be easily applied also to numerous synthesis applications. The 2DSCAR model pixel-wise synthesis is simple direct application of (1) for any 2DSCAR model.

## 2.1 Spiral Models

The 2DSCAR model’s movement  $r$  on the lattice  $I$  takes the form of circular or spiral like paths as seen in Fig. 1. The causal neighborhood  $I_r^c$  has to be transformed to be consistent for each direction in the traversed path to. The paths used can be arbitrary as long as they keep transforming the causal neighborhood into  $I_r^{cs}$  in such a way that all neighbors of a control pixel  $r$  have been visited by the model in the previous steps. We shall call all these paths as spirals further on. We present two types of paths - octagonal (Fig. 1 on the left) and a rectangular spiral (Fig. 1 - right). During our experiments they exhibited comparable results with the octagonal path being faster thanks to its consisting of fewer pixels for the same radius.

After the whole path is traversed, the parameters for the center pixel (shown as red square in Fig. 1) of the spiral are estimated. Contrary to the standard CAR model [6], since this model’s equations do not need the whole history of movement through the image but only the given one spiral, the 2DSCAR models can be easily parallelized. If the spiral paths used have circular shape, the 2DSCAR models exhibit rotational invariant properties thanks to the CAR model’s memory of all the visited pixels. The spiral neighborhood  $I_r^{cs}$  (Fig. 1 - right) is rotationally invariant only approximately. Additional contextual information can be easily incorporated if every initialization matrix  $V_0 = V_{t-1}$ , i.e., if this matrix is initialized from the previous data gathering matrix.



**Fig. 2.** Examples of images from the individual datasets. Top to bottom (rightwards): AFF (ash, black pine, fir, hornbeam, larch, mountain oak, Scots pine, spruce, Swiss stone pine, sycamore maple, beech), BarkTex (betula pendula, fagus silvatica, picea abies, pinus silvestris, quercus robur, robinia pseudacacia), Trunk12 (alder, beech, birch, ginkgo biloba, hornbeam, horse chestnut, chestnut, linden, oak, oriental plane, pine, spruce).

## 2.2 Feature Extraction

For feature extraction, we analyzed the 2DSCAR model around pixels in each spectral band with vertical and horizontal stride of 2 to speed up the computation. The following illumination invariant features originally derived for the

2DCAR model [6] were adapted for the 2DSCAR:

$$\alpha_1 = 1 + Z_r^T V_{zz}^{-1} Z_r, \quad (5)$$

$$\alpha_2 = \sqrt{\sum_r (Y_r - \hat{\gamma} Z_r)^T \lambda_r^{-1} (Y_r - \hat{\gamma} Z_r)}, \quad (6)$$

$$\alpha_3 = \sqrt{\sum_r (Y_r - \mu)^T \lambda_r^{-1} (Y_r - \mu)}, \quad (7)$$

where  $\mu$  is the mean value of vector  $Y_r$  and

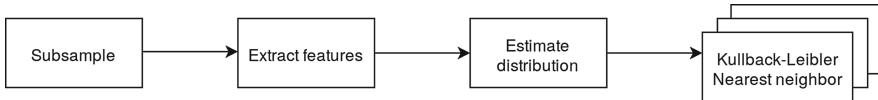
$$\lambda_{t-1} = V_{yy(t-1)} - V_{zy(t-1)}^T V_{zz(t-1)}^{-1} V_{zy(t-1)}.$$

As the texture features, we also used the estimated  $\gamma$  parameters, the posterior probability density [6]

$$p(Y_r | Y^{(r-1)}, \hat{\gamma}_{r-1}) = \frac{\Gamma(\frac{\beta(r)-\eta+3}{2})}{\Gamma(\frac{\beta(r)-\eta+2}{2}) \pi^{\frac{1}{2}} (1 + X_r^T V_{x(r-1)}^{-1} X_r)^{\frac{1}{2}} |\lambda_{(r-1)}|^{\frac{1}{2}}} \left( 1 + \frac{(Y_r - \hat{\gamma}_{r-1} X_r)^T \lambda_{(r-1)}^{-1} (Y_r - \hat{\gamma}_{r-1} X_r)}{1 + X_r^T V_{x(r-1)}^{-1} X_r} \right)^{-\frac{\beta(r)-\eta+3}{2}}, \quad (8)$$

and the absolute error of the one-step-ahead prediction

$$Abs(GE) = \left| E \left\{ Y_r | Y^{(r-1)} \right\} - Y_r \right| = |Y_r - \hat{\gamma}_{r-1} X_r|. \quad (9)$$



**Fig. 3.** Flowchart of our classification approach.

### 3 Bark Texture Recognition

To speed up the feature extraction part, we first subsample the images to the height of 300px (if the image is larger), keeping aspect ratio. This subsampling ratio depends on an application data, i.e., a compromise between the algorithm efficiency and its recognition rate. The features are then extracted as described in Sect. 2. The feature space is assumed to be approximated by the multivariate Gaussian distribution, the parameters of which are then stored for each training sample image.

$$\mathcal{N}(\theta|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} e^{(-\frac{1}{2}(\theta-\mu)^T \Sigma^{-1}(\theta-\mu))}.$$

During the classification stage, the parameters of the Gaussian distribution are estimated for the classified image as in the training step (the flowchart of our approach can be seen in Fig. 3). They are then compared with all the distributions of the training samples using the Kullback-Leibler (KL) divergence. The KL divergence is a measure of how much one probability distribution diverges from another. It is defined as:

$$D(f(x)||g(x)) \stackrel{def}{=} \int f(x) \log \frac{f(x)}{g(x)} dx.$$

For the Gaussian distribution data model, the KL divergence can be solved analytically:

$$D(f(x)||g(x)) = \frac{1}{2} \left( \log \frac{|\Sigma_g|}{|\Sigma_f|} + \text{tr}(\Sigma_g^{-1} \Sigma_f) - d + (\mu_f - \mu_g)^T \Sigma_g^{-1} (\mu_f - \mu_g) \right).$$

We use the symmetrized variant of the Kullback-Leibler divergence known as the Jeffreys divergence

$$D_s(f(x)||g(x)) = \frac{D(f(x)||g(x)) + D(g(x)||f(x))}{2}.$$

The class of the training sample with the lowest divergence from the image being recognized is then selected as the final result. The advantage of our approach is that the training database is heavily compressed through the Gaussian distribution parameters (as we extract only about 40 features, depending on the chosen neighborhood, we only need to store 40 numbers for the mean and  $40 \times 40$  numbers for the covariance matrix) and the comparison with the training database is extremely fast, enabling us to compare hundreds of thousands of image feature distributions per second on an ordinary computer.

## 4 Experimental Results

The proposed method is verified on three publicly available bark databases and our own bark dataset (not demonstrated here). Examples of images of the datasets can be seen in Fig. 2. We have used the leave-one-out approach for the classification rate estimation.

The AFF bark dataset provided by Österreichische Bundesforste, Austrian Federal Forests (AFF) [5], is a collection of the most common Austrian trees. The dataset contains 1182 bark samples belonging to 11 classes, the size of each class varying between 7 and 213 images. AFF samples are captured at different scales, and under different illumination conditions.

The Trunk12 dataset ([17], <http://www.vicos.si/Downloads/TRUNK12>) contains 393 images of tree barks belonging to 12 different trees that are found in Slovenia. The number of images per class varies between 30 and 45 images.

**Table 1.** AFF bark dataset results of the presented method (MO - Mountain oak, SP - Scots pine, SSP - Swiss stone pine, SM - Sycamore maple).

	Ash	Beech	Black pine	Fir	Horn-beam	Larch	MO	SP	Spruce	SSP	SM	Sensitivity [%]
Ash	22	0	0	1	0	0	0	0	0	0	1	91.7
Beech	0	7	0	0	0	0	0	0	0	0	0	100
B. pine	0	0	139	0	0	9	0	8	0	1	0	88.5
Fir	0	0	0	105	0	6	0	5	2	0	0	89.0
Horn.	0	0	1	0	32	0	0	0	0	0	0	97.0
Larch	0	0	6	0	0	156	0	27	0	2	0	81.7
MO	0	0	0	0	0	1	59	0	3	5	0	86.8
SP	0	0	9	1	0	28	0	142	1	0	0	78.5
Spruce	1	0	3	4	0	6	2	4	181	3	0	88.7
SSP	0	0	5	2	0	7	9	0	4	60	0	69.0
SM	1	0	0	0	3	0	3	0	0	3	2	16.7
Precision [%]	91.7	100	85.3	92.9	91.4	73.2	80.8	76.3	94.8	81.1	66.7	<b>Accuracy 83.6</b>

Bark images are captured under controlled scale, illumination and pose conditions. The classes are more homogeneous than those of AFF in terms of imaging conditions.

The BarkTex dataset [10] contains 408 samples from 6 bark classes, i.e., 68 images per class. The images have small ( $256 \times 384$ ) resolution and they have unequal natural illumination and scale.

We have achieved the accuracy of 83.6% on the AFF dataset (Table 1), 91.7% on the BarkTex database (Table 2) and 92.9% on the Trunk12 dataset (Table 3). In all the three tables, the name of the row indicates the actual tree type whereas the column indicates the predicted class. The comparison with other methods

**Table 2.** BarkTex dataset results of the presented method (BP - Betula pendula, FS - Fagus silvatica, PA - Picea abies, PS - Pinus silvestris, QR - Quercus robur, RP - Robinia pseudacacia).

	BP	FS	PA	PS	QR	RP	Sensitivity [%]
Betula pendula	64	0	0	2	2	0	94.1
Fagus silvatica	0	68	0	0	0	0	100.0
Picea abies	3	0	62	0	3	0	91.2
Pinus silvestris	0	0	1	67	0	0	98.5
Quercus robur	1	2	7	9	48	1	70.6
Robinia pseudacacia	1	0	0	1	1	65	95.6
Precision [%]	92.8	97.1	88.6	84.8	88.9	98.5	<b>Accuracy 91.7</b>

**Table 3.** Trunk12 dataset results of the presented method (A - Alder, Be - Beech, Bi - Birch, Ch - Chestnut, GB - Ginkgo biloba, H - Hornbeam, HC - Horse chestnut, L - Linden, OP - Oriental plane, S - Spruce).

	A	Be	Bi	Ch	GB	H	HC	L	Oak	OP	Pine	S	Sensitivity [%]
Alder	33	0	1	0	0	0	0	0	0	0	0	0	97.1
Beech	0	29	0	0	0	1	0	0	0	0	0	0	96.7
Birch	0	0	36	1	0	0	0	0	0	0	0	0	97.3
Chestnut	2	0	0	24	0	0	0	0	4	0	2	0	75.0
Ginkgo biloba	0	0	0	0	30	0	0	0	0	0	0	0	100
Hornbeam	0	2	0	0	0	28	0	0	0	0	0	0	93.3
Horse chestnut	0	0	1	0	0	1	27	3	0	0	1	0	81.8
Linden	0	0	0	1	0	0	4	25	0	0	0	0	83.3
Oak	1	0	0	0	0	0	0	0	29	0	0	0	96.7
Oriental plane	0	0	0	1	0	0	1	0	0	30	0	0	93.8
Pine	0	0	0	0	0	0	0	0	0	0	30	0	100
Spruce	1	0	0	0	0	0	0	0	0	0	0	44	97.8
Precision [%]	89.2	93.5	94.7	88.9	100	93.3	84.4	89.3	87.9	100	90.9	100	<b>Accuracy 92.9</b>

**Table 4.** Comparison with the state-of-the-art. ‘x’ denotes lack of results in the particular article on the given dataset.

Dataset [%]	Our results	[3]	[5]	[16]	[7]	[11]	[12]	[14]	[13]
AFF	83.6	60.5	69.7	<b>96.5</b>	-	-	-	-	-
BarkTex	<b>91.7</b>	84.6	-	-	81.4	84.7	81.4	82.1	89.6
Trunk12	<b>92.9</b>	62.8	-	-	-	-	-	-	-

is presented in Table 4. We can see that our approach vastly outperforms all compared methods on the BarkTex and Trunk12 datasets and has the second best results on the AFF dataset.

## 5 Conclusion

The presented tree bark recognition method uses an underlying descriptive textural model for the classification features and outperforms the state-of-the-art alternative methods on two public bark databases and is the second best on the AFF database. Our method is rotationally invariant, benefits from information from all spectral bands and can be easily parallelized or made fully illumination invariant. We have also executed our method without any modification on the AFF dataset’s images of needles and leaves, with results exceeding 94% accuracy. This will be a subject of our further research.

## References

1. Blaanco, L.J., Travieso, C.M., Quintero, J.M., Hernandez, P.V., Dutta, M.K., Singh, A.: A bark recognition algorithm for plant classification using a least square support vector machine. In: 2016 Ninth International Conference on Contemporary Computing, IC3, pp. 1–5, August 2016. <https://doi.org/10.1109/IC3.2016.7880233>
2. Boudra, S., Yahiaoui, I., Behloul, A.: A comparison of multi-scale local binary pattern variants for bark image retrieval. In: Battiato, S., Blanc-Talon, J., Gallo, G., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2015. LNCS, vol. 9386, pp. 764–775. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-25903-1\\_66](https://doi.org/10.1007/978-3-319-25903-1_66)
3. Boudra, S., Yahiaoui, I., Behloul, A.: Statistical radial binary patterns (SRBP) for bark texture identification. In: Blanc-Talon, J., Penne, R., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2017. LNCS, vol. 10617, pp. 101–113. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-70353-4\\_9](https://doi.org/10.1007/978-3-319-70353-4_9)
4. Chi, Z., Houqiang, L., Chao, W.: Plant species recognition based on bark patterns using novel Gabor filter banks. In: Proceedings of the 2003 International Conference on Neural Networks and Signal Processing, vol. 2, pp. 1035–1038, December 2003. <https://doi.org/10.1109/ICNNSP.2003.1281045>
5. Fiel, S., Sablatnig, R.: Automated identification of tree species from images of the bark, leaves and needles. In: 16th Computer Vision Winter Workshop, pp. 67–74. Verlag der Technischen Universität Graz (2011)
6. Haindl, M.: Visual data recognition and modeling based on local Markovian models. In: Florack, L., Duits, R., Jongbloed, G., van Lieshout, M.C., Davies, L. (eds.) Mathematical Methods for Signal and Image Analysis and Representation. CIVI, vol. 41, pp. 241–259. Springer, London (2012). [https://doi.org/10.1007/978-1-4471-2353-8\\_14](https://doi.org/10.1007/978-1-4471-2353-8_14)
7. Hoang, V.T., Porebski, A., Vandenbroucke, N., Hamad, D.: LBP histogram selection based on sparse representation for color texture classification. In: VISIGRAPP (4: VISAPP), pp. 476–483 (2017)
8. Huang, Z.K.: Bark classification using RBPNN based on both color and texture feature. *Int. J. Comput. Sci. Netw. Secur.* **6**(10), 100–103 (2006)
9. Huang, Z.K., Huang, D.S., Lyu, M.R., Lok, T.M.: Classification based on Gabor filter using RBPNN classification. In: 2006 International Conference on Computational Intelligence and Security, vol. 1, pp. 759–762. IEEE (2006)
10. Lakmann, R.: Statistische Modellierung von Farbtexturen. Ph.D. thesis (1998). <ftp://ftphost.uni-koblenz.de/de/ftp/pub/outgoing/vision/Lakman/BarkTex/>
11. Palm, C.: Color texture classification by integrative co-occurrence matrices. *Pattern Recognit.* **37**(5), 965–976 (2004)
12. Porebski, A., Vandenbroucke, N., Hamad, D.: LBP histogram selection for supervised color texture classification. In: ICIP, pp. 3239–3243 (2013)
13. Sandi, F., Douik, A.: Dominant and minor sum and difference histograms for texture description. In: 2016 International Image Processing, Applications and Systems, IPAS, pp. 1–5, November 2016. <https://doi.org/10.1109/IPAS.2016.7880136/>
14. Sandid, F., Douik, A.: Robust color texture descriptor for material recognition. *Pattern Recognit. Lett.* **80**, 15–23 (2016). <https://doi.org/10.1016/j.patrec.2016.05.010>. <http://www.sciencedirect.com/science/article/pii/S0167865516300885>
15. Song, J., Chi, Z., Liu, J., Fu, H.: Bark classification by combining grayscale and binary texture features. In: Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, pp. 450–453. IEEE (2004)



16. Sulc, M., Matas, J.: Kernel-mapped histograms of multi-scale LBPs for tree bark recognition. In: 2013 28th International Conference of Image and Vision Computing New Zealand, IVCNZ, pp. 82–87. IEEE (2013)
17. Švab, M.: Computer-vision-based tree trunk recognition (2014)
18. Wäldchen, J., Mäder, P.: Plant species identification using computer vision techniques: a systematic literature review. *Arch. Comput. Methods Eng.* **25**(2), 507–543 (2018). <https://doi.org/10.1007/s11831-016-9206-z>
19. Wan, Y.Y., et al.: Bark texture feature extraction based on statistical texture analysis. In: Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, pp. 482–485, October 2004. <https://doi.org/10.1109/ISIMP.2004.1434106>



# Dynamic Voting in Multi-view Learning for Radiomics Applications

Hongliu Cao<sup>1,2</sup>(✉), Simon Bernard<sup>2</sup>, Laurent Heutte<sup>2</sup>, and Robert Sabourin<sup>1</sup>

<sup>1</sup> LIVIA, École de Technologie Supérieure, Université du Québec,  
Montreal, Canada

caohongliu@gmail.com

<sup>2</sup> Normandie Univ, UNIROUEN, UNIHAVRE, INSA Rouen, LITIS,  
Rouen, France

**Abstract.** Cancer diagnosis and treatment often require a personalized analysis for each patient nowadays, due to the heterogeneity among the different types of tumor and among patients. Radiomics is a recent medical imaging field that has shown during the past few years to be promising for achieving this personalization. However, a recent study shows that most of the state-of-the-art works in Radiomics fail to identify this problem as a multi-view learning task and that multi-view learning techniques are generally more efficient. In this work, we propose to further investigate the potential of one family of multi-view learning methods based on Multiple Classifier Systems where one classifier is learnt on each view and all classifiers are combined afterwards. In particular, we propose a random forest based dynamic weighted voting scheme, which personalizes the combination of views for each new patient to classify. The proposed method is validated on several real-world Radiomics problems.

**Keywords:** Radiomics · Dissimilarity · Random forest  
Dynamic voting · Multi-view learning

## 1 Introduction

One of the biggest challenges of cancer treatment is the inter-tumor heterogeneity and intra-tumor heterogeneity. It demands for more personalized treatment. In Radiomics, a large amount of features from standard-of-care images obtained with CT (computed tomography), PET (positron emission tomography) or MRI (magnetic resonance imaging) are extracted to help the diagnosis, prediction or prognosis of cancer [1]. Many medical image studies like [2, 3] have already tried to use quantitative analysis before the existence of Radiomics. However, with the development of medical imaging technology and more and more available softwares allowing for more quantification and standardization, Radiomics focuses on improvements of image analysis, using an automated high-throughput extraction of large amounts of quantitative features [4]. Radiomics has the advantage of using more useful information to make optimal treatment decisions (personalized medicine) and make cancer treatment more effective and less expensive [5].

Radiomics is a promising research field for oncology, but it is also a challenging machine learning task. In the work [1], the authors identify Radiomics as a challenge in machine learning for the three following reasons: **(i) small sample size**: due to the difficulty in data sharing, most of Radiomics data sets have no more than 200 patients; **(ii) high dimensional feature space**: the feature space for Radiomics data is always very high dimensional compared to the sample size; **(iii) multiple feature groups**: different sources and different feature extractors are used in Radiomics - the most used features include tumor intensity, shape, texture, and so on [6] - and it may be hard to exploit the complementary information brought by these different views [1].

When the three challenges are encountered in a classification task, it can be seen as an HDLSS (High dimension low sample size) Multi-View learning task. Now most studies in Radiomics ignore the third challenge and propose to simply concatenate different feature groups and to use a feature selection method to reduce the dimension. However, a lot of useful information may be lost when only a small subset of features is retained [1], and the complementary information that different feature groups can offer may be ignored [7].

In contrast to the current studies that treat Radiomics data as a single-view machine learning task, we have proposed in our previous work to cope with Radiomics complexity using an HDLSS multi-view paradigm [1]: we have used a naive MCS (Multiple Classifier Systems) based method which turns out to work well for Radiomics data but not significantly better than the state of the art methods used in Radiomics. Here we want to further investigate the potential of the MCS multi-view approach. Hence we propose several less simplistic MCS based methods including static voting and dynamic voting methods to combine classification results from different views. Our main contribution in this paper is thus to propose a new dynamic voting scheme to give a personalized diagnosis (decision) from Radiomics data. This dynamic voting method is designed for small sample sized dataset like Radiomics data and uses a large number of trees in random forest to provide OOB (Out Of Bag) samples to replace the validation dataset.

The remainder of this paper is organized as follows. Related works in Radiomics and multi-view learning are discussed in Sect. 2. In Sect. 3, the proposed dynamic voting solution is introduced. Before turning to the result analysis (Sect. 5), we describe the data sets chosen in this study and provide the protocol of our experimental method in Sect. 4. We conclude and give some future works in Sect. 6.

## 2 Related Works

In the state of the art of Radiomics, groups of features are most often concatenated into a single feature vector, which results in an HDLSS machine learning problem. In order to reduce the high dimensionality, some feature selection methods are used: in the work of [6, 8], they used feature stability as a criterion for feature selection While in the work of [9], they used a SVM (Support

Vector Machine) classifier as a criterion to evaluate the predictive value of each feature for pathology and TNM clinical stage. Different filter feature selection methods have also been compared along with reliable machine learning methods to find the optimal combination [8]. Generally speaking, the embedded feature selection method SVMRFE shows good performance on different Radiomics applications [1].

A lot of studies have been done on multi-view learning and according to the work of [10], there are three main kinds of solutions: early integration, intermediate integration and late integration. Early integration concatenates information from different views together and treats it as a single-view learning task [10]. The Radiomics solutions discussed above all belong to this category. Intermediate integration combines the information from different views at the feature level to form a joint feature space. Late integration method firstly builds individual models based on separate views and then combines these models. Compared to intermediate and late integration methods, early integration always leads to high dimensional problems and the feature selection methods used in the state of the art of Radiomics can easily filter a lot of useful information.

In [1], MCS based late integration methods (with simple majority voting) have shown a big potential and a lot of flexibility on Radiomics data. In this work, to further investigate the potential of MCS for Radiomics applications, both static and dynamic combinations are tested. The intuition behind static weighted voting is that different views have different importances for a classification task. While the intuition behind proposing dynamic voting methods is that, due to the heterogeneity among patients, different patients may rely on different information sources. For example, for a patient A, there may be more useful information in one view (e.g. texture or shape features) while for a patient B, there may be more useful information in another view (e.g. intensity or wavelet features). Three dynamic integration methods were considered in the work of [11]: DS (Dynamic Selection), DV (Dynamic Voting), and DVS (Dynamic Voting with Selection). The difficulty in multi view combination is that the number of views is fixed and usually very small. In this case, dynamic selection methods may not be applicable. Hence, we focus on dynamic voting method in this work. However, traditional dynamic voting methods demand a validation dataset [12]. In Radiomics, the data size is too small to have a validation dataset. In the next section, we propose a dynamic voting method based on the random forest dissimilarity measure and the Out-Of-Bag (OOB) measure, without the need of validation dataset.

### 3 Proposed MCS Based Solutions

As explained in the Introduction, the simple MCS based late integration method used in [1] has shown a good potential for Radiomics. In this section, we use several more intelligent voting methods including static voting and dynamic voting to test if they can get significantly better.

For multi-view learning tasks, the training set  $\mathbf{T}$  is composed of  $Q$  views:  $\mathbf{T}^{(q)} = \{(\mathbf{X}_1^{(q)}, y_1), \dots, (\mathbf{X}_N^{(q)}, y_N)\}, q = 1..Q$ . Generally speaking, the MCS

based late integration method builds a classifier  $C^{(q)}$  for each view  $\mathbf{T}^{(q)}$ . During test time, for each test data  $\mathbf{X}_t$ ,  $C^{(q)}$  will predict the class label  $label_t^{(q)}$  of  $\mathbf{X}_t$ . Finally, the predicted labels from all the views  $\{label_t^{(1)}, label_t^{(2)}, \dots, label_t^{(Q)}\}$  can be combined either by majority voting or weighted voting.

Here Random forest is chosen as the classifier for each view  $\mathbf{T}^{(q)}$  because it can deal well with different data types, mixed variables and high dimensional data [1]. Random forest can also offer the OOB measure, which can be used as a measure for static weight and also to replace extra validation dataset for dynamic voting methods. In addition, random forest also provides a proximity measure, which can be used to calculate the neighborhood of a test sample [13].

Firstly, for each view  $q$ , a Random Forest  $\mathbf{H}^{(q)}$  is built with  $M$  decision trees, and is denoted as in Eq. (1):

$$\mathbf{H}(\mathbf{X}) = \{h_k(\mathbf{X}), k = 1, \dots, M\} \quad (1)$$

where  $h_k(\mathbf{X})$  is a random tree grown using bagging and random feature selection. We refer the reader to [14, 15] for more details about this procedure.

For a  $J$ -class problem with  $label_t^{(q)} = i$ , where  $i \in \{1, 2, \dots, J\}$ , a weight  $W^{(q)}$  is used for each view  $q$  (for the case of majority voting, all  $W^{(q)} = 1$ ). The final decision is made by:

$$y_t = \underset{j \in \{1, 2, \dots, J\}}{\text{Max}} \left( \sum_{q=1}^Q I(label_t^{(q)} = j) \times W^{(q)} \right) \quad (2)$$

$I()$  is an indicator function, which equals to 1 when the condition in the parenthesis is fulfilled and 0 otherwise.

### 3.1 WRF (Static Weighted Voting)

To calculate the weights for static voting, we need a measure to reflect the importance of each view to give a final decision. Usually, the prediction accuracy over a validation dataset can be used for that. However, Radiomics data have very small sample size, and it is impossible to have extra validation data. Hence we propose to use the OOB accuracy of each random forest  $\mathbf{H}^{(q)}$  as the static weight  $W^{(q)}$  for each view:

$$W_{static}^{(q)} = OOB_{accuracy}(\mathbf{H}^{(q)}) \quad (3)$$

When Bagging is used in a random forest, each bootstrap sample used to learn a single tree is typically a subset of the initial training set. This means that some of the training instances are not used in each bootstrap sample (37% in average; see [16] for more details). For a given decision tree of the forest, these instances, called the Out-of-bag (OOB) samples, can be used to estimate its accuracy. To use OOB to measure the accuracy of a random forest, the concept of sub-forest is used. When the forest size is big, all training data have a high probability to be an OOB sample at least once. Hence, for each OOB sample  $\mathbf{X}_{OOB}$ , the

trees that did not use this data as training sample are grouped together as a sub-forest  $\mathbf{H}_{sub(\mathbf{X}_{OOB})}$  (which can be seen as a representative of the complete random forest  $\mathbf{H}$ ) to give a prediction on  $\mathbf{X}_{OOB}$ . The overall accuracy of the sub-forests predictions on all OOB samples is then used as OOB accuracy for a random forest  $\mathbf{H}$ . We refer the reader to the work of [16] for further information about OOB measure.

### 3.2 GDV (Global Dynamic Voting)

In static voting, we believe that different views have different importances for classification. However, with dynamic voting, we can personalize this importance with an assumption that the importances of views are different for different patients. One easy access to this kind of “personalized” information is the prediction probability of each test sample as it shows generally how confident the classifier  $C^q$  is on the test data.

The predicted class probabilities of a test sample  $\mathbf{X}_t$  for random forest are computed as the mean predicted class probabilities of the trees in the forest. The class probabilities of a single tree is the fraction of samples of the same class in a leaf. The global weight  $W_{global}^{(q)}$  of view  $q$  for each test data  $\mathbf{X}_t$  is simply the predicted probability (posterior probability obtained from  $\mathbf{H}^{(q)}$ ) for the most confident class of random forest, which measures the overall confidence rate of label prediction based on all the training data:

$$W_{global}^{(q)} = P(label_t^{(q)} | \mathbf{X}_t, \mathbf{H}^{(q)}) \quad (4)$$

$W_{global}^{(q)}$  generally reflects how confident the classifier  $\mathbf{H}^{(q)}$  is when predicting the label of a test sample. But it also means the global measure is not very personalized. To capture more personalized information, we propose in the next subsection the local weight measure.

### 3.3 LDV (Local Dynamic Voting)

A local weight usually means the performance or confidence of a classifier in a smaller neighborhood in validation data of a test sample. It usually demands two measures: firstly, a distance measure to find the neighborhood; secondly the competence measure to evaluate the performance of the classifier in the neighborhood. RFD (random forest dissimilarity) in this work is used as a distance measure to find the neighborhood of a given test sample, while OOB measure is used to replace the validation dataset.

The RFD measure  $\mathbf{D}_{\mathbf{H}}$  is inferred from a RF classifier  $\mathbf{H}$ , learned from training data  $\mathbf{T}$ . For each tree in the forest, if two samples end in the same terminal node, their dissimilarity is 0 otherwise 1. This process goes over all trees in the forest, and the average value is the RFD value (more details are given in [1]). It can be told that compared to other dissimilarity measures, RFD takes the advantage of class information to measure the distance [1].

To calculate the local weight  $W_{local}^{(q)}$ , RFD is used to find the neighborhood  $\theta_{\mathbf{X}}$  of each test instance  $\mathbf{X}$  by choosing the most  $n_{neighbor}$  similar instances in training data. The OOB measure over  $\theta_{\mathbf{X}}$  is then used to calculate the local weight. Unlike in the work of [11] using OOB to measure the individual tree accuracy, here OOB is used to measure the performance of the RF classifier. With  $\theta_{\mathbf{X}}$ , the local weight can be easily calculated with OOB measure:

$$W_{local}^{(q)} = OOB_{accuracy}(\mathbf{H}^{(q)}, \theta_{\mathbf{X}}) \quad (5)$$

The idea of local weight here is similar to OLA (Overall Local Accuracy) used in dynamic selection [12]. There are two main differences: firstly, LDV uses the random forest dissimilarity as a distance measure which carries both feature information and class label information while OLA uses Euclidean distance which may suffer from the concentration of pairwise distance [17] in high dimensional space; secondly, OLA requires a validation dataset while LDV does not.

### 3.4 GLDV (Global and Local Dynamic Voting)

From the previous two subsections, we can see that  $W_{global}^{(q)}$  uses global information from all training data and measures the confidence of the classifier. But it has also the risk of being too generalized and lacks of personalized information. On the other hand,  $W_{local}^{(q)}$  uses information on the neighborhood of the test sample to give a more personalized measure which can better represent the heterogeneity among cancer patients but may lose the global vision at the same time. Hence we propose a measure that takes both measures into account.

With each  $\mathbf{H}^{(q)}$ , the global weight  $W_{global}^{(q)}$  and the local weight  $W_{local}^{(q)}$  are calculated respectively and the combined weight  $W_{GL}^{(q)}$  is calculated by taking advantage of both global and local information together:

$$W_{GL}^{(q)} = W_{global}^{(q)} \times W_{local}^{(q)} \quad (6)$$

The reason why we choose to multiply global weight and local weight for deriving a combined weight, is that, as it is explained previously,  $W_{global}$  lacks personalized information, but it can be counter-balanced by  $W_{local}$  to give more preference in some situations. For example, when  $W_{global}^{(q)}$  agrees with  $W_{local}^{(q)}$  on a particular view  $q$ , if both weights are small, then  $W_{GL}^{(q)}$  becomes even smaller as we do not have confidence on this view; if both weights get bigger and bigger, then  $W_{GL}^{(q)}$  gets closer and closer to both weights, especially local weight. On the contrary, when  $W_{global}^{(q)}$  disagrees with  $W_{local}^{(q)}$ , it is hard to make a decision with a disagreement (as we need prior knowledge to decide to choose global or local weight); hence we penalize  $W_{GL}^{(q)}$  as long as there is a disagreement ( $W_{GL}^{(q)}$  is smaller than 0.5) but still with a preference to  $W_{local}^{(q)}$ .

## 4 Experiments

In this study, we use several publicly available Radiomics datasets. A general description of all datasets can be found in Table 1 where  $IR$  stands for the imbalance ratio of the dataset. More details about these datasets can be found in the work of [18].

**Table 1.** Overview of each dataset.

	#Features	#Samples	#Views	#Classes	IR
nonIDH1	6746	84	5	2	3
IDHcodel	6746	67	5	2	2.94
lowGrade	6746	75	5	2	1.4
progression	6746	75	5	2	1.68

The main objective of the experiment is to compare the state of the art Radiomics methods to static and dynamic voting methods. In total six methods are compared: one state of the art Radiomics method, i.e. SVMRFE; two static weighting methods, i.e. MVRF (combines RF results with majority voting as in [1]) and WRF (combines RF results with weights as in Sect. 3.1, the weights are the OOB accuracy of each  $\mathbf{H}^{(a)}$ ); three dynamic weighted voting methods, i.e. GDV, LDV and GLDV as described in the previous section.

For the two dynamic voting methods that use local weights, LDV and GLDV, the neighborhood size  $n_{neighbor}$  is set to 7 according to the work of [12]. For SVMRFE, the number of selected features is defined as in [1] according to the experiments of [19] and a Random forest classifier is then built on the selected features. For all random forest classifiers, the tree number is set to 500 while the other parameters are set to the default values given by the Scikit-Learn package for Python.

Similar to our previous work [1, 7], a stratified repeated random sampling approach was used to achieve a robust estimate of the performance. The stratified random splitting procedure is repeated 10 times, with 50% sample rate in each subset. In order to compare the methods, the mean and standard deviations of accuracy are evaluated over 10 runs.

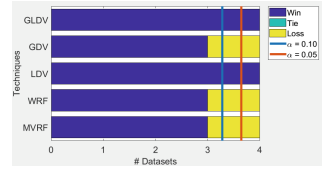
## 5 Results

The results of mean accuracies, along with the corresponding standard deviation, over the 10 repetitions are shown in Table 2. GDV and the two static voting methods have almost the same results over the four datasets, but these results are different from the two dynamic weighted voting methods LDV and GLDV. It is not surprising that there is no difference between MVRF and WRF because the datasets we use in this work have only five views, which means that there is



**Table 2.** Experiment results with 50% training data 50% test data for Radiomics data

Dataset	SVMRFE +RF	MVRF	WRF	GDV	LDV	GLDV
nonIDH1	76.28% ±4.39	82.79% ±2.37	82.79% ±2.37	82.79% ±2.37	76.98% ±1.93	77.44% ±2.33
IDHcodel	73.23% ±5.50	76.76% ±2.06	76.76% ±2.06	76.76% ±2.06	74.11% ±1.17	74.41% ±1.34
lowGrade	62.55% ±3.36	64.41% ±3.76	64.41% ±3.76	64.41% ±3.76	64.41% ±3.45	66.05% ±3.32
progression	62.36% ±3.73	61.31% ±4.25	61.31% ±4.25	61.57% ±4.27	62.63% ±4.37	62.89% ±4.62
Average rank	5.250	3.250	3.250	2.875	3.875	<b>2.500</b>

**Fig. 1.** Pairwise comparison between MCS solutions and SVM-RFE. The vertical lines illustrate the critical values considering a confidence level  $\alpha = \{0.10, 0.05\}$ .

no situation like even votes (the worst case would be 3 against 2). Hence as long as there is no extremely big difference among performance of different views, the two static voting methods should have similar results. And the result of GDV confirms our assumption in the previous section that the global weight alone does not contain a lot of personalized information. We can also see that there is a benefit of combining global and local weights as the performance of GLDV is always better than LDV. From the average ranking value, it can be told that the best method is the proposed GLDV method, followed by GDV. The state of the art solution SVMRFE is ranked at the last place.

To see more clearly the difference between MCS based methods and SVM-RFE, a pairwise analysis based on the Sign test is computed on the number of wins, ties and losses as in the work of [12]. Figure 1 shows that, when compared to SVMRFE, only the proposed methods LDV and GLDV are significantly better than SVMRFE with  $\alpha = 0.10$  and  $0.05$ . These results show that the MCS based late integration methods can also be significantly better than the state-of-art Radiomics solutions.

When we compare GDV, LDV and GLDV, it can be seen that for nonIDH1 and IDHCodel data, the performance of GLDV is between LDV and GDV (LDV is the worst while GDV is the best). However for the two other datasets, GLDV is always better than both LDV and GDV, which means that for different datasets, the best combination of LDV and GDV should be different. To further study the preference of global weight  $W_{global}^{(q)}$  and local weight  $W_{local}^{(q)}$  for different datasets, a new combination is formed as:

$$W_{GLnew}^{(q)} = (W_{global}^{(q)})^{1-a} \times (W_{local}^{(q)})^a \quad (7)$$

From Eq. 7 it can be told that when  $a = 1$ , the combination is only affected by local accuracy while when  $a = 0$  the combination is only affected by global accuracy. The results of  $W_{GLnew}^{(q)}$  are shown in Table 3, from which we can confirm our conclusion that for IDHCodel1 and nonIDH data, they get better results

**Table 3.** The results of new combinations  $W_{GLnew}^{(q)}$  with different  $a$  value.

Dataset	a = 0 (GDV)	a = 0.1	a = 0.2	a = 0.3	a = 0.4	a = 0.5	a = 0.6	a = 0.7	a = 0.8	a = 0.9	a = 1 (LDV)
nonIDH	<b>82.79%</b> ±2.37	<b>82.79%</b> ±2.37	<b>82.79%</b> ±2.37	82.32% ±2.13	81.16% ±3.02	80.23% ±2.80	79.99% ±3.15	79.30% ±2.42	77.90% ±2.38	77.44% ±2.33	76.97% ±1.93
IDHCodel1	<b>76.76%</b> ±2.06	<b>76.76%</b> ±2.06	<b>76.76%</b> ±2.06	75.88% ±1.76	75.58% ±1.34	75.29% ±1.44	75.29% ±1.44	75.29% ±1.95	75.00% ±1.97	75.00% ±1.97	74.41% ±1.34
lowGrade	64.41% ±3.75	64.41% ±3.75	64.41% ±3.75	<b>64.65%</b> ±3.57	64.41% ±3.45	64.41% ±3.45	<b>64.65%</b> ±3.72	64.18% ±4.18	63.48% ±3.75	63.48% ±3.45	63.95% ±3.64
progression	61.57% ±4.27	61.57% ±4.27	61.84% ±3.57	62.10% ±3.56	62.36% ±3.91	62.10% ±4.43	62.36% ±4.41	<b>63.42%</b> ±4.62	62.89% ±4.77	62.89% ±4.77	62.36% ±4.56

when they use more global weight. For lowGrade and progression data, they get better results when they use more local weight.

In general, all MCS based late integration methods are better than feature selection methods. Majority voting is simple and efficient. GLDV is only better than majority voting on two datasets. But LDV and GLDV are preferable for Radiomics applications in the following three ways: (i) they give different weights of each view to each test sample, so that each test sample uses a different combination of classifiers to give a personalized decision; (ii) they are significantly better than the state of art work in Radiomics; (iii) the performance of GLDV can be further improved by adjusting the proportion of local weight and global weight. Note that other parameters like the neighborhood size can also be adjusted to optimize the performance. Compared to static voting, the disadvantage of dynamic voting is that it is more complex and less efficient.

## 6 Conclusions

In the state of art works of Radiomics, most studies used feature selection methods as a solution for the HDLSS problem. In this work, we have treated Radiomics as a multi-view learning problem and investigated the potential of MCS based late integration methods, proposed earlier in [1]. In particular, we have investigated some dynamic voting based MCS methods, that can give each patient a personalized prediction by dynamically integrating the classification result from each view. We believe these methods have a great potential and can significantly outperform early integration methods that make use of feature selection in the concatenated feature space.

To confirm our hypothesis, a representative early integration method, five MCS methods including three dynamic voting methods and two static voting methods, have been compared on four Radiomics datasets. We conclude from our experiments that all MCS based late integration methods are generally better than the state of art Radiomics solution, but only LDV and GLDV are significantly better, which shows the potential of MCS based late integration methods of being a better solution than the state-of-art Radiomics solutions.

**Acknowledgment.** This work is part of the DAISI project, co-financed by the European Union with the European Regional Development Fund (ERDF) and by the Normandy Region.

## References

1. Cao, H., Bernard, S., Heutte, L., Sabourin, R.: Dissimilarity-based representation for radiomics applications. ICPRAI 2018, [arXiv:1803.04460](https://arxiv.org/abs/1803.04460) (2018)
2. Sorensen, L., Shaker, S.B., De Bruijne, M.: Quantitative analysis of pulmonary emphysema using local binary patterns. *IEEE Trans. Med. Imaging* **29**(2), 559–569 (2010)
3. Sluimer, I., Schilham, A., Prokop, M., Van Ginneken, B.: Computer analysis of computed tomography scans of the lung: a survey. *IEEE Trans. Med. Imaging* **25**(4), 385–405 (2006)
4. Lambin, P., et al.: Radiomics: extracting more information from medical images using advanced feature analysis. *Eur. J. Cancer* **48**(4), 441–446 (2012)
5. Kumar, V., et al.: Radiomics: the process and the challenges. *Magn. Reson. Imaging* **30**(9), 1234–1248 (2012)
6. Aerts, H., et al.: Decoding tumour phenotype by noninvasive imaging using a quantitative radiomics approach. *Nat. Commun.* **5**, 1–8 (2014)
7. Cao, H., Bernard, S., Heutte, L., Sabourin, R.: Improve the performance of transfer learning without fine-tuning using dissimilarity-based multi-view learning for breast cancer histology images. ICIAR 2018, [arXiv:1803.11241](https://arxiv.org/abs/1803.11241) (2018)
8. Parmar, C., Grossmann, P., Rietveld, D., Rietbergen, M.M., Lambin, P., Aerts, H.J.: Radiomic machine-learning classifiers for prognostic biomarkers of head and neck cancer. *Front. Oncol.* **5**, 272 (2015)
9. Song, J., et al.: Non-small cell lung cancer: quantitative phenotypic analysis of ct images as a potential marker of prognosis. *Sci. Rep.* **6**, 38282 (2016)
10. Serra, A., Fratello, M., Fortino, V., Raiconi, G., Tagliaferri, R., Greco, D.: MVDA: a multi-view genomic data integration methodology. *BMC Bioinform.* **16**(1), 261 (2015)
11. Tsymbal, A., Pechenizkiy, M., Cunningham, P.: Dynamic integration with random forests. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 801–808. Springer, Heidelberg (2006). [https://doi.org/10.1007/11871842\\_82](https://doi.org/10.1007/11871842_82)
12. Cruz, R.M., Sabourin, R., Cavalcanti, G.D.: Dynamic classifier selection: recent advances and perspectives. *Inf. Fusion* **41**, 195–216 (2018)
13. Tsymbal, A., Pechenizkiy, M., Cunningham, P., Puuronen, S.: Dynamic integration of classifiers for handling concept drift. *Inf. Fusion* **9**(1), 56–68 (2008)
14. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)
15. Biau, G., Scornet, E.: A random forest guided tour. *Test* **25**(2), 197–227 (2016)
16. Breiman, L.: Out-of-bag estimation. Technical report 513, University of California, Department of Statistics, Berkeley (1996)
17. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the surprising behavior of distance metrics in high dimensional space. In: Van den Bussche, J., Vianu, V. (eds.) ICDDT 2001. LNCS, vol. 1973, pp. 420–434. Springer, Heidelberg (2001). [https://doi.org/10.1007/3-540-44503-X\\_27](https://doi.org/10.1007/3-540-44503-X_27)
18. Zhou, H., et al.: MRI features predict survival and molecular markers in diffuse lower-grade gliomas. *Neuro-Oncology* **19**(6), 862–870 (2017)
19. Bolón-Canedo, V., Sánchez-Maróño, N., Alonso-Betanzos, A.: A review of feature selection methods on synthetic data. *Knowl. Inf. Syst.* **34**(3), 483–519 (2013)



# Iterative Deep Subspace Clustering

Lei Zhou<sup>1</sup>, Shuai Wang<sup>1</sup>, Xiao Bai<sup>1(✉)</sup>, Jun Zhou<sup>2</sup>, and Edwin Hancock<sup>3</sup>

<sup>1</sup> School of Computer Science and Engineering and Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing, China  
{leizhou,wangshuai,baixiao}@buaa.edu.cn

<sup>2</sup> School of Information and Communication Technology, Griffith University,  
Brisbane, Queensland, Australia  
jun.zhou@griffith.edu.au

<sup>3</sup> Department of Computer Science, University of York, York, UK  
edwin.hancock@york.ac.uk

**Abstract.** Recently, deep learning has been widely used for subspace clustering problem due to the excellent feature extraction ability of deep neural network. Most of the existing methods are built upon the auto-encoder networks. In this paper, we propose an iterative framework for unsupervised deep subspace clustering. In our method, we first cluster the given data to update the subspace ids, and then update the representation parameters of a Convolutional Neural Network (CNN) with the clustering result. By iterating the two steps, we can obtain not only a good representation for the given data, but also more precise subspace clustering result. Experiments on both synthetic and real-world data show that our method outperforms the state-of-the-art on subspace clustering accuracy.

**Keywords:** Subspace clustering · Unsupervised deep learning  
Convolutional Neural Network

## 1 Introduction

In many computer vision applications, such as face recognition [5, 13], texture recognition [16] and motion segmentation [7], visual data can be well characterized by subspaces. Moreover, the intrinsic dimension of high-dimensional data is often much smaller than the ambient dimension [26]. This has motivated the development of subspace clustering techniques which simultaneously cluster the data into multiple subspaces and also locate a low-dimensional subspace for each class of data.

Many subspace clustering algorithms have been developed during the past decade, including algebraic [27], iterative [1], statistical [22], and spectral clustering methods [2–4, 7, 13, 15–17, 31, 32]. Among these approaches, spectral clustering methods have been intensively studied due to their simplicity, theoretical soundness, and empirical success. These methods are based on the self-expressiveness property of data lying in a union of subspaces. This states that

each point in a subspace can be written as a linear combination of the remaining data points in that subspace. One of the typical method falling into this category is sparse subspace clustering (SSC) [7]. SSC uses the  $\ell_1$  norm to encourage the sparsity of the self-representation coefficient matrix.

Although those subspace clustering methods have shown encouraging performance, we observe that they suffer from the following limitations. First, most subspace clustering methods learn data representation via shallow models which may not capture the complex latent structure of big data. Second, the methods require to access the whole data set as the dictionary, and thus making difficulty in handling large scale and dynamic data set. To solve these problems, we believe that deep learning could be an effective solution thanks to its outperforming representation learning capacity and fast inference speed. In fact, [19, 29, 30] have very recently proposed to learn representation for clustering using deep neural networks. However, most of them do not work in an end-to-end manner which however is generally believed to be the major factor for the success of deep learning [6, 12].

In this work, we aim to address subspace clustering and representation learning on unlabeled images in a unified framework. It is a natural idea to leverage cluster ids of images as supervisory signals to learn representations and in turn the representations would be beneficial to subspace clustering. Specifically, we first cluster the given data to update the subspace ids, and then update the representation parameters of a Convolutional Neural Network (CNN) with the clustering result. By iterating the two steps, we can obtain not only a good representation for the given data, but also more precise subspace clustering result.

The main contributions of this paper are as follows:

1. We propose a simple but effective end-to-end learning framework to jointly learn deep representations and subspace clustering result;
2. We formulate the joint learning in a recurrent framework, where merging operations of subspace clustering are expressed as a forward pass, and representation learning of CNN as a backward pass;
3. Experimental results on both synthetic data and real world public datasets show that our method leads to a improvement in the clustering accuracy compared with the state-of-the-art methods.

## 2 Related Work

### 2.1 Subspace Clustering

The past decade saw an upsurge of subspace clustering methods with various applications in computer vision, e.g. motion segmentation, face clustering image processing, multi-view analysis, and video analysis. Particularly, among these works, spectral clustering based methods have achieved state-of-the-art results. The key of these methods is to learn a satisfactory affinity matrix  $C$  in which  $C_{i,j}$  denotes the similarity between the  $i$ -th and the  $j$ -th sample. Given a data matrix  $X = [x_i \in \mathbb{R}^D]_{i=1}^N$  that contains  $N$  data points drawn from  $n$  subspaces

$\{S_i\}_{i=1}^n$ . SSC [7] aims to find a sparse representation matrix  $C$  showing the mutual similarity of the points, i.e.,  $X = XC$ . Since each point in  $S_i$  can be expressed in terms of the other points in  $S_i$ , such a sparse representation matrix  $C$  always exists. The SSC algorithm finds  $C$  by solving the following optimization problem:

$$\min_C \|C\|_1 \quad \text{s.t. } X = XC, \text{diag}(C) = 0, \quad (1)$$

where  $\text{diag}(C) = 0$  eliminates the trivial solution. Different works adopt different regularization on  $C$  and three of them are most popular, i.e.  $\ell_1$ -norm based sparsity [7, 8], nuclear-norm based low rankness [13, 25, 28], and Frobenius norm based sparsity [18, 21].

## 2.2 Deep Learning

During the past several years, most existing subspace clustering methods focus on how to learn a good data representation that is beneficial to discover the inherent clusters. As the most effective representation learning technique, deep learning has been extensively studied for various applications, especially, in the scenario of supervised learning [10, 11]. In contrast, only a few of works have devoted to unsupervised scenario which is one of major challenges faced by deep learning [6, 12]. In work [24], the authors adopted the auto-encoder network to clustering. Specifically, Tian et al. [24] proposed a novel graph clustering approach in the sparse auto-encoder framework. Furthermore, Peng et al. [19] presented a deep subspAce clusteRing with sparsiTY prior, termed as PARTY, by combining the deep neural network and sparsity information of original data to perform subspace clustering. This framework achieved a satisfactory performance while extracting low-dimensional feature in the unsupervised learning.

## 3 Proposed Method

### 3.1 Problem Statement

Let  $X = [x_i \in \mathbb{R}^D]_{i=1}^N \in \mathbb{R}^{D \times N}$  be a collection of data points drawn from different subspaces. The goal of subspace clustering is to find the segmentation of the points according to the subspaces. Based on the self-expressiveness property of data lying in a union of subspaces, i.e., each point in a subspace can be written as a linear combination of the remaining points in that subspace, we can obtain points lying in the same subspace by learning the sparsest combination. Therefore, we need to learn a sparse self-representation coefficient matrix  $C$ , where  $X = XC$ , and  $C_{ij} = 0$  if the  $i$ -th and  $j$ -th data points are from different subspaces.

Our iterative method aims to learn data representations and subspace clustering result simultaneously. We first utilize sparse subspace clustering algorithm to cluster the given data to update the subspace ids, and then update the representation parameters of a Convolutional Neural Network with the clustering

result. By iterating the two steps, we can obtain not only a good representation for the given data, but also more precise subspace clustering result.

**Notation.** We denote the data matrix as  $X = \{x_i \in \mathbb{R}^D\}_{i=1}^N$  that contains  $N$  data points drawn from  $n$  subspaces  $\{S_i\}_{i=1}^n$ . The cluster labels for these data are  $y = \{y_1, \dots, y_N\}$ .  $\theta$  are the CNN parameters, based on which we obtain deep representations  $\widehat{X} = \{\widehat{x}_1, \dots, \widehat{x}_N\}$  from  $X$ . We add a superscript  $t$  to  $\{\theta, X, \widehat{X}, y\}$  to refer to their states at timestep  $t$ .

### 3.2 An Iterative Method

We propose a iterative framework to combine the subspace clustering and representation learning processes.

As shown in Fig. 1, at the timestep  $t$ , we first cluster the data representation  $\widehat{X}^{t-1}$  to get the subspace cluster labels  $y^t$ . Then fed  $X$  and  $y^t$  into the CNN to get representations  $\widehat{X}^t$ . Hence, at timestep  $t$

$$y^t = SSC(\widehat{X}^{t-1}) \quad (2)$$

$$\{\widehat{X}^t, \theta^t\} = f(X|y^t) \quad (3)$$

where SSC is the classical sparse subspace clustering method [7], and  $f$  is a function to extract deep representations  $\widehat{X}^t$  for input  $X$  using the CNN trained with  $y^t$ .

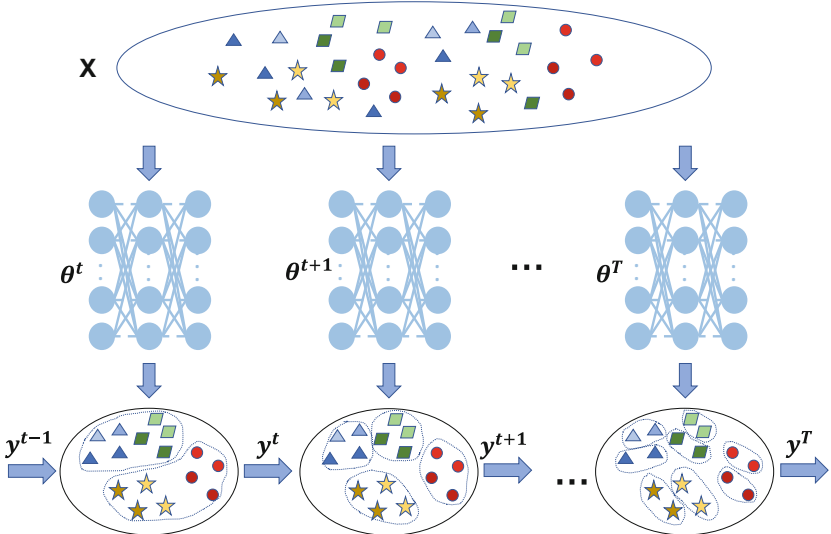
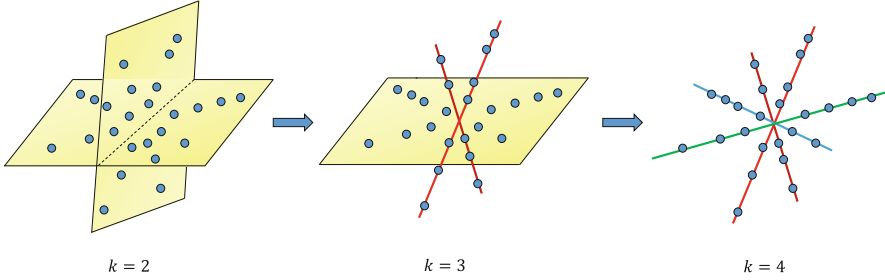


Fig. 1. The process of our proposed iterative method for deep subspace clustering.



**Fig. 2.** An illustration of our updating process for subspace clustering.

Since the initialized clustering result may be not reliable. We start with an initial over-clustering. As shown in Fig. 2, we first cluster the data into 2 subspaces, then increase the cluster number  $k$  and iterate until reaching a stopping criterion.

In our iterative framework, we accumulate the losses from all timesteps, which is formulated as

$$L(y^1, \dots, y^T; \theta^1, \dots, \theta^T | X) = \sum_{t=1}^T L^t(y^t, \theta^t | X) \quad (4)$$

$$L^t(y^t, \theta^t | X) = \|\widehat{X}^{t-1} - \widehat{X}^{t-1} C^t\|_F^2 + \lambda \|C^t\|_1 \quad (5)$$

We assume the number of desired clusters is  $n$ . Then we can build up an iterative process with  $T = n - 1$  timesteps. We first cluster the data into 2 subspaces as initial clusters. Given these initial clusters, our method learns deep representations for the data. Then for the new data representations, we cluster them into 3 subspaces and learn update representations with the update subspace labels. As summarized in Algorithm 1, we iterate this process until the number of clusters reaches  $n$ . In each iterative period, we perform forward and backward passes to update  $y$  and  $\theta$  respectively. Specifically, in the forward pass,

---

**Algorithm 1.** Iterative method for deep subspace clustering

---

**Input:** A set of data points  $X = \{x_i\}_{i=1}^N$ , the number of subspaces  $n$ .

**Steps:**

1.  $t = 1$ .
2. Initialize  $y$  by clustering the data into 2 clusters.
3. Initialize  $\theta$  by training CNN with the initialize  $y$ .
4. Update  $y^t$  to  $y^{t+1}$  by increasing one cluster.
5. Update  $\theta^t$  to  $\theta^{t+1}$  by training CNN.
6.  $t = t + 1$ .
7. Iterate step 4 to step 6 until  $t = n$ .

**Output:** Final data representations and subspace clustering result.

---



we increase one cluster at each timestep. In the backward pass, we run about 20 epochs to update  $\theta$ , and the affinity matrix  $C$  is also updated based on the new representation.

## 4 Experiments

We have conducted three sets of experiments on both real and synthetic datasets to verify the effectiveness of the proposed methods. Several state-of-the-art or classical subspace clustering methods were taken as the baseline algorithms. These included sparse subspace clustering (SSC) [7], low-rank representation (LRR) [13], least squares regression (LSR) [14], smooth representation clustering (SMR) [9], thresholding ridge regression (TRR) [20], Kernel sparse subspace clustering (KSSC) [15] and deep subspace clustering with sparsity prior (PARTY) [19].

**Evaluation Criteria:** we used the clustering accuracy to evaluate the performance of the subspace clustering methods, which is calculated as

$$\text{clustering accuracy} = \frac{\# \text{ of correctly classified points}}{\text{total } \# \text{ of points}} \times 100$$

### 4.1 Synthetic Data

To verify the effectiveness of our method in the condition that each subspace with different number of data points, we ran experiments on synthetic data. Following [31], we randomly generated  $n = 5$  subspaces, each of dimension  $d = 6$  in an ambient space of dimension  $D = 9$ . Each subspace contains  $N_i$  data points randomly generated on the unit sphere, where  $N_i \in \{100, 200, 500, 800, 1000, 1500, 2000\}$ , so the number of points  $N \in \{500, 1000, 2500, 4000, 5000, 7500, 10000\}$ . For our iterative method, the total timestep  $T = n - 1 = 4$ , i.e., iterating with four times. With different number of sample points in each subspace, we conducted experiments on all methods and report the clustering accuracy in Table 1.

As shown in Table 1, the clustering accuracy of our method has an improvement compared with state-of-the-art methods. Our method also outperforms the deep learning based subspace clustering method [19] by the iterative rule. From Table 1, it is also clear that when the dataset size increases, our method achieves more significant improvement than the other methods.

### 4.2 Face Clustering

As subspaces are commonly used to capture the appearance of faces under varying illuminations, we test the performance of our method on face clustering with the CMU PIE database [23]. The CMU PIE database contains 41,368 images of 68 people under 13 different poses, 43 different illumination conditions, and 4 different expressions. In our experiment, we used the face images in five near frontal poses (P05, P07, P09, P27, P29). Then each people has 170

**Table 1.** The subspace clustering accuracy on synthetic data.

Method	Number of data points in each subspace						
	100	200	500	800	1000	1500	2000
SSC [7]	0.9415	0.9402	0.9386	0.9374	0.9283	0.9214	0.9105
LRR [13]	0.9312	0.9323	0.9284	0.9236	0.9165	0.9102	0.9042
LSR [14]	0.9347	0.9315	0.9241	0.9179	0.9124	0.9085	0.9012
SMR [9]	0.9431	0.9418	0.9347	0.9285	0.9221	0.9120	0.9116
TRR [20]	0.9613	0.9585	0.9562	0.9523	0.9485	0.9436	0.9414
KSSC [15]	0.9213	0.9322	0.9315	0.9236	0.9152	0.9103	0.9021
PARTY [19]	0.9605	0.9601	0.9589	0.9537	0.9503	0.9479	0.9453
<b>Ours</b>	<b>0.9721</b>	<b>0.9754</b>	<b>0.9713</b>	<b>0.9685</b>	<b>0.9642</b>	<b>0.9612</b>	<b>0.9604</b>

face images under different illuminations and expressions. Each image was manually cropped and normalized to a size of  $32 \times 32$  pixels. In each experiment, we randomly picked  $n \in \{5, 10, 20, 30, 40, 50, 60\}$  individuals to investigate the performance of the proposed method. Then, for our method, the total timestep  $T = n - 1 = \{4, 9, 19, 29, 39, 49, 59\}$ . For different number of objects  $n$ , we randomly chose  $n$  people with 10 trials and took all the images of them as the subsets to be clustered. Then we conducted experiments on all 10 subsets and report the average clustering accuracy with a different number of objects in Table 2.

In our experiment, the data size is in the range of  $N \in \{850, 1700, 3400, 5100, 6800, 8500, 10200\}$ , corresponding to 5–60 objects per face. As shown in Table 2, the clustering accuracy of other methods degrades drastically when  $N$  increases. But our iterative method only has a slight degrades when  $N$  increases. Also, our method achieves the best clustering accuracy among the existing methods.

**Table 2.** The subspace clustering accuracy on the CMU PIE database.

Method	Different number of objects						
	5	10	20	30	40	50	60
SSC [7]	0.9247	0.8925	0.8431	0.8345	0.8237	0.8035	0.7912
LRR [13]	0.9453	0.8827	0.8386	0.8274	0.8175	0.8062	0.8022
LSR [14]	0.9214	0.9052	0.8523	0.8365	0.8021	0.7924	0.7763
SMR [9]	0.9315	0.9106	0.8732	0.8512	0.8228	0.8112	0.8052
TRR [20]	<b>0.9735</b>	0.9605	0.9454	0.9243	0.9174	0.9012	0.8835
KSSC [15]	0.9621	0.9532	0.9201	0.9023	0.8837	0.8413	0.8105
PARTY [19]	0.9655	0.9529	0.9358	0.9125	0.9015	0.8921	0.8845
<b>Ours</b>	0.9675	<b>0.9612</b>	<b>0.9546</b>	<b>0.9465</b>	<b>0.9384</b>	<b>0.9235</b>	<b>0.9068</b>

### 4.3 Handwritten Digit Clustering

Database of handwritten digits is also widely used in subspace learning and clustering. We test the proposed method on handwritten digit clustering with the MNIST dataset. This dataset contains 10 clusters, including handwritten digits 0–9. Each cluster contains 6,000 images for training and 1,000 images for testing, with a size of  $28 \times 28$  pixels in each image. We used all the 70,000 handwritten digit images for subspace clustering. Different from the experimental settings for face clustering, we fixed the number of clusters  $n = 10$  and chose different number of data points for each cluster with 10 trials. Each cluster contains  $N_i$  data points randomly chosen from corresponding 7,000 images, where  $N_i \in \{50, 100, 500, 1000, 2000, 5000, 7000\}$ , so that the number of points  $N \in \{500, 1000, 5000, 10000, 20000, 50000, 70000\}$ . Then we applied all methods on this dataset for comparison. For our models, the total timestep  $T = n - 1 = 9$ , i.e., iterating with 9 times. The average clustering accuracy with different number of data points are shown in Table 3.

It can be seen that the average clustering accuracy of our method outperforms the state-of-the-art methods, which indicates the effectiveness of the iterative rule based deep subspace clustering method.

**Table 3.** The subspace clustering accuracy on the MNIST dataset.

Method	Number of data points in each cluster						
	50	100	500	1000	2000	5000	7000
SSC [7]	0.8336	0.8245	0.8014	0.7735	0.7412	0.7104	0.6857
LRR [13]	0.8575	0.8514	0.8278	0.8012	0.7756	0.7317	0.7031
LSR [14]	0.8521	0.8462	0.8213	0.8016	0.7721	0.7316	0.7041
SMR [9]	0.8362	0.8325	0.8102	0.7836	0.7524	0.7231	0.7014
TRR [20]	0.9028	0.8978	0.8621	0.8345	0.8012	0.7754	0.7371
KSSC [15]	0.8721	0.8634	0.8412	0.8155	0.7936	0.7515	0.7205
PARTY [19]	0.9132	0.9105	0.8923	0.8731	0.8516	0.8213	0.8031
<b>Ours</b>	<b>0.9231</b>	<b>0.9225</b>	<b>0.9105</b>	<b>0.9056</b>	<b>0.8934</b>	<b>0.8865</b>	<b>0.8735</b>

## 5 Conclusion

We have presented an iterative framework for unsupervised deep subspace clustering. We first cluster the given data to update the subspace ids, and then update the representation parameters of a Convolutional Neural Network with the clustering result. By iterating the two steps, we can obtain not only a good representation for the given data, but also more precise subspace clustering result. Thanks to the superiority of the deep convolutional neural network in representation learning capacity, the subspace clustering accuracy of our iterative

method achieves significant improvement compared with several state-of-the-art approaches (SSC, LRR, LSR, SMR, TRR, KSSC and PARTY). Experimental results on both synthetic and real-world public data show the superiority of our method. Moreover, by experiments designed with different conditions (different number of data points in each cluster and different number of clusters), it is obvious that our method is more scalable for different applications.

In the future work, we aim to solve the efficiency problem. Since the efficiency of our iterative method suffers for the desired number of clusters, i.e., the number of iterations.

**Acknowledgement.** This work was supported by the National Natural Science Foundation of China project no. 61772057, in part by Beijing Natural Science Foundation project no. 4162037, and the support funding from State Key Lab. of Software Development Environment.

## References

1. Agarwal, P.K., Mustafa, N.H.: K-means projective clustering. In: Symposium on Principles of Database Systems, pp. 155–165 (2004)
2. Bai, X., Yang, H., Zhou, J., Ren, P., Cheng, J.: Data-dependent hashing based on p-stable distribution. *IEEE Trans. Image Process.* **23**(12), 5033–5046 (2014)
3. Bai, X., Yan, C., Yang, H., Bai, L., Zhou, J., Hancock, E.R.: Adaptive hash retrieval with kernel based similarity. *Pattern Recogn.* **75**, 136–148 (2018)
4. Bai, X., Zhang, H., Zhou, J.: VHR object detection based on structural feature extraction and query expansion. *IEEE Trans. Geosci. Remote Sens.* **52**(10), 6508–6520 (2014)
5. Basri, R., Jacobs, D.W.: Lambertian reflectance and linear subspaces. *IEEE Trans. Pattern Anal. Mach. Intell.* **25**(2), 218–233 (2003)
6. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(8), 1798–1828 (2013)
7. Elhamifar, E., Vidal, R.: Sparse subspace clustering: algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(11), 2765–2781 (2013)
8. Feng, J., Lin, Z., Xu, H., Yan, S.: Robust subspace segmentation with block-diagonal prior. In: *Computer Vision and Pattern Recognition*, pp. 3818–3825 (2014)
9. Hu, H., Lin, Z., Feng, J., Zhou, J.: Smooth representation clustering. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3834–3841 (2014)
10. Hu, J., Lu, J., Tan, Y.P.: Discriminative deep metric learning for face verification in the wild. In: *Computer Vision and Pattern Recognition*, pp. 1875–1882 (2014)
11. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: *International Conference on Neural Information Processing Systems*, pp. 1097–1105 (2012)
12. Lecun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553), 436 (2015)
13. Liu, G., Lin, Z., Yan, S., Sun, J., Yu, Y., Ma, Y.: Robust recovery of subspace structures by low-rank representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(1), 171–184 (2013)
14. Lu, C.-Y., Min, H., Zhao, Z.-Q., Zhu, L., Huang, D.-S., Yan, S.: Robust and efficient subspace segmentation via least squares regression. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) *ECCV 2012*. LNCS, vol. 7578, pp. 347–360. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-33786-4\\_26](https://doi.org/10.1007/978-3-642-33786-4_26)

15. Patel, V.M., Vidal, R.: Kernel sparse subspace clustering. In: International Conference on Image Processing, pp. 2849–2853 (2014)
16. Peng, C., Kang, Z., Cheng, Q.: Subspace clustering via variance regularized ridge regression. In: Computer Vision and Pattern Recognition (2017)
17. Peng, C., Kang, Z., Yang, M., Cheng, Q.: Feature selection embedded subspace clustering. *IEEE Sign. Process. Lett.* **23**(7), 1018–1022 (2016)
18. Peng, X., Lu, C., Zhang, Y., Tang, H.: Connections between nuclear-norm and frobenius-norm-based representations. *IEEE Trans. Neural Netw. Learn. Syst.* **PP**(99), 1–7 (2015)
19. Peng, X., Xiao, S., Feng, J., Yau, W.Y., Yi, Z.: Deep subspace clustering with sparsity prior. In: International Joint Conference on Artificial Intelligence, pp. 1925–1931 (2016)
20. Peng, X., Yi, Z., Tang, H.: Robust subspace clustering via thresholding ridge regression. In: AAAI Conference on Artificial Intelligence, pp. 3827–3833 (2015)
21. Peng, X., Yu, Z., Yi, Z., Tang, H.: Constructing the l2-graph for robust subspace learning and subspace clustering. *IEEE Trans. Cybern.* **47**(4), 1053 (2016)
22. Rao, S.R., Tron, R., Vidal, R., Ma, Y.: Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In: Computer Vision and Pattern Recognition, pp. 1–8 (2008)
23. Sim, T., Baker, S., Bsat, M.: The CMU pose, illumination, and expression (PIE) database of human faces. Technical report, CMU-RI-TR-01-02, Pittsburgh, PA, January 2001
24. Tian, F., Gao, B., Cui, Q., Chen, E., Liu, T.Y.: Learning deep representations for graph clustering. In: Twenty-Eighth AAAI Conference on Artificial Intelligence, pp. 1293–1299 (2014)
25. Vidal, R., Favaro, P.: Low rank subspace clustering (LRSC). *Pattern Recogn. Lett.* **43**(1), 47–61 (2014)
26. Vidal, R.: Subspace clustering. *IEEE Signal Process. Mag.* **28**(2), 52–68 (2011)
27. Vidal, R., Ma, Y., Sastry, S.: Generalized principal component analysis (GPCA). *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(12), 1945–1959 (2005)
28. Xiao, S., Tan, M., Xu, D., Dong, Z.Y.: Robust kernel low-rank representation. *IEEE Trans. Neural Netw. Learn. Syst.* **27**(11), 2268–2281 (2016)
29. Xie, J., Girshick, R., Farhadi, A.: Unsupervised deep embedding for clustering analysis. In: International Conference on Machine Learning, pp. 478–487 (2016)
30. Yang, J., Parikh, D., Batra, D.: Joint unsupervised learning of deep representations and image clusters. In: Computer Vision and Pattern Recognition, pp. 5147–5156 (2016)
31. You, C., Robinson, D., Vidal, R.: Scalable sparse subspace clustering by orthogonal matching pursuit. In: Computer Vision and Pattern Recognition, pp. 3918–3927 (2016)
32. Zhang, H., Bai, X., Zhou, J., Cheng, J., Zhao, H.: Object detection via structural feature selection and shape model. *IEEE Trans. Image Process.* **22**(12), 4984–4995 (2013)



# A Scalable Spectral Clustering Algorithm Based on Landmark-Embedding and Cosine Similarity

Guangliang Chen<sup>(✉)</sup>

Department of Mathematics and Statistics, San José State University,  
San José, CA 95192, USA  
guangliang.chen@sjsu.edu

**Abstract.** We extend our recent work on scalable spectral clustering with cosine similarity (ICPR'18) to other kinds of similarity functions, in particular, the Gaussian RBF. In the previous work, we showed that for sparse or low-dimensional data, spectral clustering with the cosine similarity can be implemented directly through efficient operations on the data matrix such as elementwise manipulation, matrix-vector multiplication and low-rank SVD, thus completely avoiding the weight matrix. For other similarity functions, we present an embedding-based approach that uses a small set of landmark points to convert the given data into sparse feature vectors and then applies the scalable computing framework for the cosine similarity. Our algorithm is simple to implement, has clear interpretations, and naturally incorporates an outliers removal procedure. Preliminary results show that our proposed algorithm yields higher accuracy than existing scalable algorithms while running fast.

## 1 Introduction

Owing to the pioneering work [10, 12, 15] at the beginning of the century, spectral clustering has emerged as a very promising clustering approach. The fundamental idea is to construct a weighted graph on the given data and use spectral graph theory [5] to embed data into a low dimensional space (spanned by the top few eigenvectors of the weight matrix), where the data is clustered via the  $k$ -means algorithm. We display the Ng-Jordan-Weiss (NJW) version of spectral clustering [12] in Algorithm 1 and shall focus on this algorithm in this paper. For other versions of spectral clustering such as the Normalized Cut [15], or for a tutorial on spectral clustering, we refer the reader to [9].

Due to the nonlinear embedding by the eigenvectors, spectral clustering can easily adapt to non-convex geometries and accurately separate non-intersecting shapes. As a result, it has been successfully used in many applications, e.g., document clustering, image segmentation, and community detection in social networks. Nevertheless, the applicability of spectral clustering has been limited to small data sets because of its high computational complexity associated to the weight matrix  $\mathbf{W}$  (defined in Algorithm 1): For a given data set of  $n$  points,

---

**Algorithm 1.** (review) Spectral Clustering by Ng, Jordan, and Weiss (NIPS 2001)

---

**Input:** Data points  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ , # clusters  $k$ , tuning parameter  $\sigma$

**Output:** A partition of given data into  $k$  clusters  $C_1, \dots, C_k$

1: Construct the pairwise similarity matrix

$$\mathbf{W} = (w_{ij}) \in \mathbb{R}^{n \times n}, \quad w_{ij} = \begin{cases} \exp(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}), & \text{if } i \neq j \\ 0, & \text{if } i = j \end{cases}$$

2: Form a diagonal matrix  $\mathbf{D} \in \mathbb{R}^{n \times n}$  with entries  $\mathbf{D}_{ii} = \sum_j w_{ij}$ .

3: Use  $\mathbf{D}$  to normalize  $\mathbf{W}$  by the formula  $\widetilde{\mathbf{W}} = \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$ .

4: Find the top  $k$  eigenvectors of  $\widetilde{\mathbf{W}}$  (corresponding to the largest  $k$  eigenvalues) and stack them into a matrix  $\mathbf{V} = [\mathbf{v}_1 | \dots | \mathbf{v}_k] \in \mathbb{R}^{n \times k}$ .

5: Rescale the row vectors of  $\mathbf{V}$  to have unit length and use the  $k$ means algorithm to group them into  $k$  clusters.

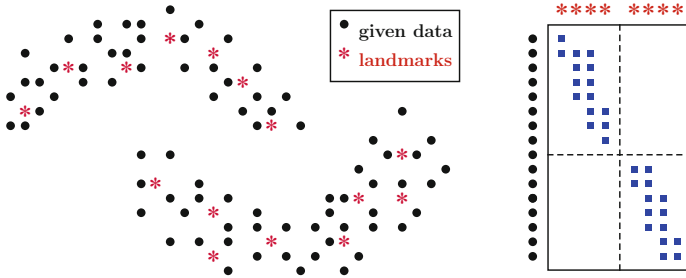
---

the storage requirement for  $\mathbf{W}$  is  $\mathcal{O}(n^2)$  while the time complexity for computing its eigenvectors is  $\mathcal{O}(n^3)$ .

Consequently, there has been considerable work on fast, approximate spectral clustering for large data sets [2–4, 8, 11, 14, 16–19]. Interestingly, the majority of them use a selected *landmark set* to help reduce the computational complexity. Specifically, they first find a small set of  $\ell \ll n$  data representatives (called *landmarks*) from the given data and then construct a similarity matrix  $\mathbf{A} \in \mathbb{R}^{n \times \ell}$  between the given data and selected landmarks (see Fig. 1), which is much smaller than  $\mathbf{W}$ . Afterwards, different algorithms use the matrix  $\mathbf{A}$  in different ways for clustering the given data. For example, the column-sampling spectral clustering (cSPEC) algorithm [18] regards  $\mathbf{A}$  as a column-sampled version of  $\mathbf{W}$  and uses the left singular vectors of  $\mathbf{A}$  to approximate the eigenvectors of  $\mathbf{W}$ , while the landmark-based spectral clustering (LSC) algorithm [2] interprets the rows of  $\mathbf{A}$  as approximate sparse representations of the original data and applies spectral clustering accordingly to group them into  $k$  clusters.

In our recent work [3] we introduced a scalable implementation of various spectral clustering algorithms [6, 12, 15] in the special setting of cosine similarity by exploiting the product form of the weight matrix. We showed that if the data is large in size ( $n$ ) but has some sort of low dimensional structure – either of low dimension ( $d$ ) or being sparse (e.g. as a document-term matrix), then one can perform spectral clustering with cosine similarity solely based on three kinds of efficient operations on the data matrix: elementwise manipulation, matrix-vector multiplication, and low-rank SVD. As a result, the algorithm enjoys a linear complexity in the size of the data.

In this work we extend the methodology in [3] to handle other kinds of similarity functions, in particular, the Gaussian radial basis function (RBF). Like most existing approaches, we also start by selecting a small subset of landmark points from the given data and constructing an affinity matrix  $\mathbf{A}$  between the given data and the selected landmarks (see Fig. 1). However, we interpret the



**Fig. 1.** Illustration of landmark-based methods. Left: given data and selected landmarks; Right: the similarity matrix between them, with the blue squares indicating the largest entries in each row (which correspond to the nearest landmark points). Here, both the given data and the landmarks have been sorted according to the true clusters. (Color figure online)

rows of  $\mathbf{A}$  as an embedding of the given data into some feature space ( $\mathbb{R}^\ell$ ), and expect the different clusters to be separated by angle in the feature space. Accordingly, we apply the scalable implementation of spectral clustering with the cosine similarity [3] to the rows of  $\mathbf{A}$  in order to cluster the original data.

The rest of the paper is organized as follows. In Sect. 2 we review our previous work in the special setting of cosine similarity. We then present in Sect. 3 a new scalable spectral clustering framework for general similarity measures. Experiments are conducted in Sect. 4 to numerically test our algorithm. Finally, in Sect. 5, we conclude the paper while pointing out some future directions.

**Notation.** Vectors are denoted by boldface lowercase letters (e.g.,  $\mathbf{a}$ ,  $\mathbf{b}$ ). The  $i$ th element of  $\mathbf{a}$  is written as  $a_i$  or  $\mathbf{a}(i)$ . We denote the constant vector of one (in column form) as  $\mathbf{1}$ , with its dimension implied by the context. Matrices are denoted by boldface uppercase letters (e.g.,  $\mathbf{A}$ ,  $\mathbf{B}$ ). The  $(i, j)$  entry of  $\mathbf{A}$  is denoted by  $a_{ij}$  or  $\mathbf{A}(i, j)$ . The  $i$ th row of  $\mathbf{A}$  is denoted by  $\mathbf{A}(i, :)$  while its columns are written as  $\mathbf{A}(:, j)$ , as in MATLAB. We use  $\mathbf{I}$  to denote the identity matrix (with its dimension implied by the context).

## 2 Recent Work

In this section we review our recent work on scalable spectral clustering with the cosine similarity [3], which does not need to compute the  $n \times n$  weight matrix but instead operates directly on the data matrix.

Let  $\mathbf{X} \in \mathbb{R}^{n \times d}$  be a data set of  $n$  points in  $\mathbb{R}^d$  to be divided into  $k$  disjoint subsets by spectral clustering with the cosine similarity. We assume that  $\mathbf{X}$  is large in size ( $n$ ) but satisfies one of the following low-dimension conditions:

- (a)  **$\mathbf{d}$  is also large but  $\mathbf{X}$  is a sparse matrix.** This is the typical setting of documents clustering [1] in which  $\mathbf{X}$  represents a document-term frequency matrix under the bag-of-words model.



- (b)  $\mathbf{d} \ll \mathbf{n}$  (but  $\mathbf{X}$  can be a full matrix). This is the case for many image data sets, for instance, the MNIST handwritten digits<sup>1</sup> ( $n = 70,000, d = 784$ ).

The two conditions together are fairly general, because for high dimensional non-sparse data, one can apply principal component analysis (PCA) to embed them into several hundred dimensions (such that the condition  $d \ll n$  is true).

For the sake of calculating cosine similarity, we assume that the given data points have nonnegative coordinates (which is true for document and image data) and are normalized to have unit  $L_2$  norm. It follows that the cosine similarity matrix is given by

$$\mathbf{W} = \mathbf{X}\mathbf{X}^T - \mathbf{I} \in \mathbb{R}^{n \times n}. \quad (1)$$

To carry out a scalable implementation of spectral clustering with the above weight matrix, we first calculate the degree matrix  $\mathbf{D} = \text{diag}(\mathbf{W}\mathbf{1})$  as follows (which avoids the expensive matrix multiplication  $\mathbf{X}\mathbf{X}^T$ ):

$$\mathbf{D} = \text{diag}((\mathbf{X}\mathbf{X}^T - \mathbf{I})\mathbf{1}) = \text{diag}(\mathbf{X}(\mathbf{X}^T\mathbf{1}) - \mathbf{1}). \quad (2)$$

Next, to find the top  $k$  eigenvectors  $\tilde{\mathbf{U}}$  of the symmetric normalization  $\tilde{\mathbf{W}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$  (but without being given  $\mathbf{W}$ ), we write

$$\tilde{\mathbf{W}} = \mathbf{D}^{-1/2}(\mathbf{X}\mathbf{X}^T - \mathbf{I})\mathbf{D}^{-1/2} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^T - \tilde{\mathbf{D}}^{-1}, \quad (3)$$

where  $\tilde{\mathbf{X}} = \mathbf{D}^{-1/2}\mathbf{X}$ . Note that the matrix  $\tilde{\mathbf{X}}$  has the same size and sparsity pattern with  $\mathbf{X}$ . If  $\mathbf{D}^{-1}$  has a constant diagonal, then the eigenvectors of  $\tilde{\mathbf{W}}$  coincide with the left singular vectors of  $\tilde{\mathbf{X}}$ , in which case we can compute  $\tilde{\mathbf{U}}$  directly based on the rank- $k$  SVD of  $\tilde{\mathbf{X}}$ . In practical settings when  $\mathbf{D}^{-1}$  does not have a constant diagonal, we propose to remove from the given data a fraction of points that correspond to the smallest diagonal entries of  $\mathbf{D}$  to make  $\mathbf{D}^{-1}$  approximately constant diagonal and correspondingly use the left singular vectors of  $\tilde{\mathbf{X}}$  to approximate the eigenvectors of  $\tilde{\mathbf{W}}$ . Such a technique can also be justified from an outliers removal perspective, since the diagonal entries of  $\mathbf{D}$  measure the connectivity of the vertices on the graph. By removing low-connectivity points which tend to be outliers, we can improve the clustering accuracy and meanwhile obtain robust statistics of the underlying clusters.

We summarize the above steps in Algorithm 2, which was first introduced in [3].

### 3 Proposed Algorithm

In this section we introduce a new scalable spectral clustering algorithm that works for any similarity function. However, for the exposition of ideas, we shall focus on the Gaussian similarity:

$$\kappa_G(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/(2\sigma^2)}, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d \quad (4)$$

<sup>1</sup> Available at <http://yann.lecun.com/exdb/mnist/>.

---

**Algorithm 2.** (review) Scalable Spectral Clustering with Cosine Similarity

---

**Input:** Data matrix  $\mathbf{X} \in \mathbb{R}^{n \times d}$  (sparse or of moderate dimension, with  $L_2$  normalized rows), # clusters  $k$ , fraction of outliers  $\alpha$

**Output:** Clusters  $C_1, \dots, C_k$  and a set of outliers  $C_0$

- 1: Calculate the degree matrix  $\mathbf{D} = \text{diag}(\mathbf{X}(\mathbf{X}^T \mathbf{1}) - \mathbf{1})$  and remove the bottom  $(100\alpha)\%$  of the input data (with lowest degrees) as outliers (stored in  $C_0$ ).
  - 2: For the remaining data, compute  $\tilde{\mathbf{X}} = \mathbf{D}^{-1/2} \mathbf{X}$  and find its top  $k$  left singular vectors  $\tilde{\mathbf{U}}$  by rank- $k$  SVD.
  - 3: Normalize the rows of  $\tilde{\mathbf{U}}$  to have unit length and apply  $k$ -means to find  $k$  clusters  $C_1, \dots, C_k$ .
- 

where  $\sigma$  is a parameter to be tuned by the user. When applied to a data set  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ , this function generates an  $n \times n$  symmetric similarity matrix

$$\mathbf{W} = (w_{ij}), \quad w_{ij} = \kappa_G(\mathbf{x}_i, \mathbf{x}_j). \quad (5)$$

It does not have a product form as in the case of cosine similarity, so we cannot directly employ the computing techniques presented in Sect. 2.

To deal with the Gaussian similarity, we regard  $\mathbf{W}$  not as a weight matrix, but as a feature matrix:

$$\mathbf{x}_i \in \mathbb{R}^d \mapsto \mathbf{W}(i, :) \in \mathbb{R}^n, \quad 1 \leq i \leq n. \quad (6)$$

That is, each  $\mathbf{x}_i$  is mapped to a feature vector (i.e., the  $i$ th row of  $\mathbf{W}$ ) containing its similarity with every point in the whole data set, but having large similarities only with points from the same cluster.<sup>2</sup> Collectively, different clusters in the original space are mapped to (nearly) orthogonal locations in the feature space, so that the original proximity-based clustering problem becomes an angle-based one. This suggests that we can *in principle* apply spectral clustering with the cosine similarity to the row vectors of  $\mathbf{W}$  to cluster the original data.

To practically realize the above idea, we observe that many of the columns of  $\mathbf{W}$  (as features) carry very similar discriminatory information and thus are highly redundant. Accordingly, we propose to sample a fraction of them for forming a reduced feature matrix and expect the sampled columns to still contain sufficient discriminatory information. We also point out that the columns of  $\mathbf{W}$  are defined by isotropic Gaussian distributions at different data points  $\mathbf{x}_j$ :

$$\mathbf{W}(:, j) = \left( e^{-\frac{\|\mathbf{x}_1 - \mathbf{x}_j\|^2}{2\sigma^2}}, \dots, e^{-\frac{\|\mathbf{x}_n - \mathbf{x}_j\|^2}{2\sigma^2}} \right)^T, \quad 1 \leq j \leq n. \quad (7)$$

Thus, sampling columns can be thought of as selecting a collection of small, round Gaussian distributions (to represent the data distribution). Under such a new perspective, we can relax the Gaussian centers  $\{\mathbf{x}_j\}$  to be any kind of data

---

<sup>2</sup> This is similarity-based feature representation. Note that there is also work on dissimilarity representation [7, 13].

representatives (e.g., local centroids). We denote such broadly defined Gaussian centers by  $\mathbf{c}_1, \dots, \mathbf{c}_\ell$  (for some  $\ell \ll n$ ) and call them *landmark points*.

Two simple ways of choosing the landmark points are *uniform sampling* and *k-means sampling*. The former approach samples uniformly at random a subset of the data as the Gaussian centers while the latter applies *k-means* to partition the data into many small clusters and uses their centroids as the landmark points. The first sampling approach is obviously faster but the second may yield much better landmark points.

Regardless of the sampling method, we use the selected landmark points to form a feature matrix  $\mathbf{A} \in \mathbb{R}^{n \times \ell}$ :

$$\mathbf{A}(i, j) = \kappa_G(\mathbf{x}_i, \mathbf{c}_j) = e^{-\frac{\|\mathbf{x}_i - \mathbf{c}_j\|^2}{2\sigma^2}}. \quad (8)$$

Since  $\ell \ll n$ , the rows of  $\mathbf{A}$  could already be provided directly to Algorithm 2 as input data. To improve efficiency and possibly also accuracy, we propose the following enhancements before we apply Algorithm 2:

- **Sparsification:** Due to fast decay of the Gaussian function, we expect each row  $\mathbf{A}(i, :)$  to have only a few large entries (which correspond to the nearest landmark points of  $\mathbf{x}_i$ ). To promote such sparsity, we fix an integer  $s \geq 1$  and truncate each row of  $\mathbf{A}$  by keeping only its  $s$  largest entries (the rest are set to zero). This results in a sparse feature matrix with a moderate dimension, which is computationally very efficient.
- **Column normalization.** After the row-sparsification step, we normalize the columns of  $\mathbf{A}$  to have unit  $L_2$  norm in order to give all landmarks equal importance. This also seems to match the  $L_2$  row normalization performed afterwards for calculating the cosine similarity.

*Remark 1.* The LSC algorithm [2] uses the same sparsification step on the matrix  $\mathbf{A}$ , but based on a sparse coding perspective. It then performs  $L_1$  row normalization on  $\mathbf{A}$ , followed by square-root  $L_1$  column normalization, which is quite different from what we proposed above.

We now summarize all the steps of our scalable implementation of spectral clustering with the Gaussian similarity in Algorithm 3.

---

**Algorithm 3.** (proposed) Scalable Spectral Clustering with Gaussian Similarity

---

**Input:** Data  $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ , # clusters  $k$ , landmark sampling method, # landmark points  $\ell$ , # nearest landmark points  $s$ , % outliers  $\alpha$ , tuning parameter  $\sigma$

**Output:** Clusters  $C_1, \dots, C_k$  and a set of outliers  $C_0$

- 1: Select  $\ell$  landmark points  $\{\mathbf{c}_j\}$  by the given sampling method.
  - 2: Compute the feature matrix  $\mathbf{A} \in \mathbb{R}^{n \times \ell}$  via (8), and apply the two enhancements in turn:  $s$ -sparsification of rows and  $L_2$  normalization along columns.
  - 3: Apply Alg. 2 with  $\mathbf{A}$  as input data along with parameters  $k$  and  $\alpha$  to partition the data into  $k$  clusters  $\{C_i\}$  and an outliers set  $C_0$ .
-

Finally, we mention the complexity of Algorithm 3. The storage requirement is  $\mathcal{O}(n\ell)$  (with uniform sampling) or  $\mathcal{O}(nd)$  (with  $k$ -means sampling). The computational complexity of Algorithm 3 with uniform sampling is  $\mathcal{O}(nlk)$ , as it takes  $\mathcal{O}(n\ell)$  time to compute the feature matrix  $\mathbf{A}$  and  $\mathcal{O}(nlk)$  time to apply Algorithm 2 to cluster the row vectors of  $\mathbf{A}$  (which have a moderate dimension  $\ell$ ). If  $k$ -means sampling is used instead, then it requires  $\mathcal{O}(ndl)$  time additionally.

## 4 Experiments

We conduct numerical experiments to test our proposed algorithm (i.e., Algorithm 3) against several existing scalable methods: cSPEC [18], LSC [2], and the  $k$ -means-based approximate spectral clustering algorithm (KASP) [19], which aggressively reduces the given data to a small set of centroids found by  $k$ -means.

We choose six benchmark data sets - *usps*, *pendigits*, *letter*, *protein*, *shuttle*, *mnist* - from the LIBSVM website<sup>3</sup> for our study; see Table 1 for their summary information. These data sets are originally partitioned into training and test parts for classification purposes, but for each data set we have merged the two parts together for our unsupervised setting.

**Table 1.** Data sets used in our study.

Dataset	#pts( $n$ )	#dims( $d$ )	#classes( $k$ )	$\ell = \sqrt{nk}/2$
usps	9,298	256	10	153
pendigits	10,992	16	10	166
letter	20,000	16	26	361
protein	24,387	357	3	136
shuttle	58,000	9	7	319
mnist	70,000	784	10	419

We implemented all the methods (except LSC<sup>4</sup>) in MATLAB 2016b and conducted the experiments on a compute server with 48 GB of RAM and 2 CPUs with 12 total cores. In order to have fair comparisons, we use the same parameter values and landmark sets (whenever they are shared) for the different algorithms. In particular, we fix  $\ell = \frac{1}{2}\sqrt{nk}$  for all methods<sup>5</sup> (see the last column of Table 1 for their actual values) and  $s = 6$  (for LSC and our algorithm only; the other two methods KASP and cSPEC do not need this parameter). For our proposed algorithm and LSC, we implement both the uniform and  $k$ -means

<sup>3</sup> <https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/>.

<sup>4</sup> Code available at <http://www.cad.zju.edu.cn/home/dengcai/Data/Clustering.html>.

<sup>5</sup> This empirical rule is derived as  $\ell = \frac{1}{2} \cdot \sqrt{\frac{n}{k}} \cdot k = \frac{1}{2}\sqrt{nk}$ , with the intuition that the value of  $\ell$  should be proportional to both the (average) cluster size and number of clusters. For the data sets in Table 1, such an  $\ell$  is always a few hundred.

sampling methods for landmark selection, but for each of KASP and cSPEC, we implement only one of the two sampling methods according to their original designs: cSPEC<sup>(n)</sup> (only uniform sampling) and KASP (only  $k$ means sampling). Lastly, for the proposed algorithm, we fix the  $\alpha$  parameter to 0.01 in all cases, and set the tuning parameter  $\sigma$  as half of the average distance between each given data point and its  $s$ th nearest neighbor in the landmark set.

We evaluate the different algorithms in terms of clustering accuracy and CPU time (both averaged over 50 replications), with the former being calculated by first finding the best match between the output cluster labels and the ground truth and then computing the fraction of correctly assigned labels.

We report the results in Tables 2 and 3. Regarding the clustering accuracy, observe that our proposed algorithm performed the best in the most cases with each kind of sampling, and was very close to the best methods in all other cases. Regarding running time, all the methods are more or less comparable, with our proposed method being the fastest in the case of uniform sampling and KASP being the fastest when  $k$ -means sampling is used. Overall, our proposed algorithm obtained very competitive and stable accuracy while running fast.

We next study the sensitivity of the parameter  $s$  by varying its value from 2 to 12 continuously for LSC and our proposed method (with both sampling

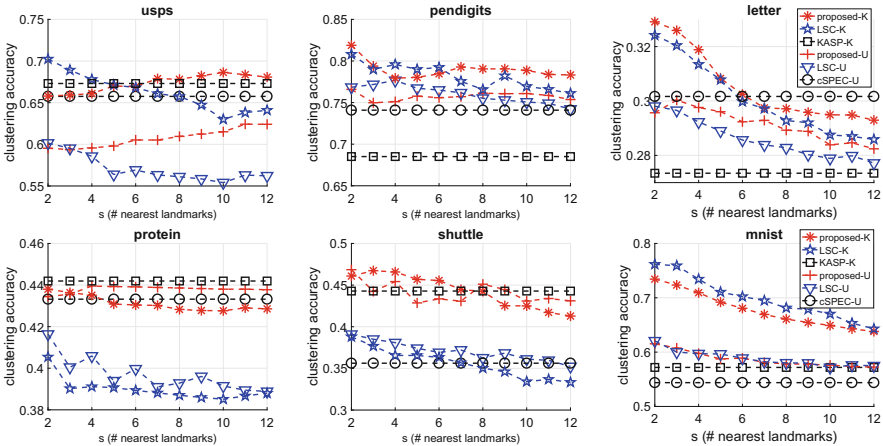
**Table 2.** Mean and standard deviation (over 50 trials) of the clustering accuracy (%) obtained by the various methods on the benchmark data sets in Table 1.

	Uniform sampling			$k$ -means sampling		
	Proposed	LSC	cSPEC	Proposed	LSC	KASP
usps	61.0±1.8	56.1±3.9	<b>65.8±4.4</b>	<b>67.8±2.3</b>	65.7±5.1	67.3±4.1
pendigits	<b>76.1±3.5</b>	75.5±4.0	74.1±4.8	<b>79.1±5.2</b>	76.6±4.0	68.5±5.2
letter	28.9±1.3	28.3±1.5	<b>30.2±1.4</b>	<b>29.7±1.3</b>	29.3±1.2	27.3±1.1
protein	<b>43.9±0.8</b>	39.3±2.1	43.3±0.3	42.8±0.7	38.7±1.1	<b>44.2±1.7</b>
shuttle	<b>45.1±0.9</b>	36.3±4.7	35.6±7.7	44.2±8.2	35.0±4.7	<b>44.3±7.8</b>
mnist	57.8±1.6	<b>58.0±2.9</b>	54.4±2.2	66.1±2.3	<b>68.1±3.8</b>	57.2±2.3

**Table 3.** Average CPU time (in seconds) used by the various methods.

	Uniform sampling			$k$ -means sampling		
	Proposed	LSC	cSPEC	Proposed	LSC	KASP
usps	3.7	5.8	5.6	4.3	5.7	1.2
pendigits	3.0	3.9	5.5	3.4	4.6	0.9
letter	20.5	16.7	42.3	22.3	19.5	3.2
protein	2.5	5.7	4.7	5.5	8.9	3.7
shuttle	13.4	7.1	11.6	15.4	10.8	5.2
mnist	23.1	23.5	44.1	42.4	44.9	26.7

schemes). For each data set, we fix  $\ell$  to the value shown in Table 1. This experiment is also repeated 50 times in order to compute the average accuracy and time (for different values of  $s$ ); see Fig. 2. In general, increasing the value of  $s$  tends to decrease the accuracy (with some exceptions). Observe also that the proposed method lies at (or stays close to) the top of every plot for many values of  $s$ , demonstrating its stable and competitive accuracy.



**Fig. 2.** Effects of the parameter  $s$ . In all plots the color and symbol of each method is fixed, so only one legend box is displayed in each row (the suffixes ‘-U’ and ‘-K’ denote the uniform and  $k$ -means sampling schemes, respectively). Since cSPECK and KASP do not need this parameter, we have plotted them as constant lines. (Color figure online)

## 5 Conclusions and Future Work

We presented a new scalable spectral clustering approach based on a landmark-embedding technique and our recent work on scalable spectral clustering with the cosine similarity. Our implementation is simple, fast, and accurate, and is naturally combined with an outliers removal procedure. Preliminary experiments conducted in this paper demonstrate competitive and stable performance of the proposed algorithm in terms of both clustering accuracy and speed.

We plan to continue the research along the following directions: (1) Our previous work on scalable spectral clustering with the cosine similarity actually covers the Normalized Cut algorithm [15] and Diffusion Maps [6], but they have been left out due to space constraints. Our next step is to implement them in the case of the Gaussian similarity. (2) In this paper we fix the number of landmarks by the formula  $\ell = \frac{1}{2}\sqrt{nk}$ , and did not conduct a sensitivity study of this parameter. We will run some experiments in this aspect and report the results in a future publication. (3) Our methodology actually assumes a mixture of Gaussians model for each cluster (when the Gaussian affinity is used), which

opens a door for probabilistic analysis of the algorithm. We plan to study the theoretical properties of the proposed algorithm in the near future.

**Acknowledgments.** We thank the anonymous reviewers for their helpful feedback. This work was motivated by a project sponsored by Verizon Wireless, which had the goal of grouping customers based on similar profile characteristics. G. Chen was supported by the Simons Foundation Collaboration Grant for Mathematicians.

## References

1. Aggarwal, C.C., Zhai, C.: A survey of text clustering algorithms. In: Aggarwal, C., Zhai, C. (eds.) *Mining Text Data*, pp. 77–128. Springer, Boston (2012). [https://doi.org/10.1007/978-1-4614-3223-4\\_4](https://doi.org/10.1007/978-1-4614-3223-4_4)
2. Cai, D., Chen, X.: Large scale spectral clustering via landmark-based sparse representation. *IEEE Trans. Cybern.* **45**(8), 1669–1680 (2015)
3. Chen, G.: Scalable spectral clustering with cosine similarity. In: *Proceedings of the 24th International Conference on Pattern Recognition (ICPR)*, Beijing, China (2018)
4. Jain, S., Munos, R., Stephan, F., Zeugmann, T. (eds.): *ALT 2013. LNCS (LNAI)*, vol. 8139. Springer, Heidelberg (2013). <https://doi.org/10.1007/978-3-642-40935-6>
5. Chung, F.R.K.: *Spectral graph theory*. In: *CBMS Regional Conference Series in Mathematics*, vol. 92. AMS (1996)
6. Coifman, R., Lafon, S.: Diffusion maps. *Appl. Comput. Harmonic Anal.* **21**(1), 5–30 (2006)
7. Duin, R., Pekalska, E.: The dissimilarity space: bridging structural and statistical pattern recognition. *Pattern Recogn. Lett.* **33**(7), 826–832 (2012)
8. Fowlkes, C., Belongie, S., Chung, F., Malik, J.: Spectral grouping using the Nyström method. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(2), 214–225 (2004)
9. von Luxburg, U.: A tutorial on spectral clustering. *Stat. Comput.* **17**(4), 395–416 (2007)
10. Meila, M., Shi, J.: A random walks view of spectral segmentation. In: *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics* (2001)
11. Moazzen, Y., Tasdemir, K.: Sampling based approximate spectral clustering ensemble for partitioning data sets. In: *Proceedings of the 23rd International Conference on Pattern Recognition* (2016)
12. Ng, A., Jordan, M., Weiss, Y.: On spectral clustering: analysis and an algorithm. *Adv. Neural Inf. Process. Syst.* **14**, 849–856 (2001)
13. Pekalska, E., Duin, R.: *The Dissimilarity Representation for Pattern Recognition*. World Scientific, Singapore (2005)
14. Pham, K., Chen, G.: Large-scale spectral clustering using diffusion coordinates on landmark-based bipartite graphs. In: *Proceedings of the 12th Workshop on Graph-based Natural Language Processing (TextGraphs-2012)*, pp. 28–37. Association for Computational Linguistics (2018)
15. Shi, J., Malik, J.: Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(8), 888–905 (2000)
16. Tasdemir, K.: Vector quantization based approximate spectral clustering of large datasets. *Pattern Recogn.* **45**(8), 3034–3044 (2012)

17. Wang, L., Leckie, C., Kotagiri, R., Bezdek, J.: Approximate pairwise clustering for large data sets via sampling plus extension. *Pattern Recogn.* **44**, 222–235 (2011)
18. Wang, L., Leckie, C., Ramamohanarao, K., Bezdek, J.: Approximate spectral clustering. In: Theeramunkong, T., Kijssirikul, B., Cercone, N., Ho, T.-B. (eds.) *PAKDD 2009*. LNCS (LNAI), vol. 5476, pp. 134–146. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-01307-2\\_15](https://doi.org/10.1007/978-3-642-01307-2_15)
19. Yan, D., Huang, L., Jordan, M.: Fast approximate spectral clustering. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 907–916 (2009)



# **Deep Learning and Neural Networks**



# On Fast Sample Preselection for Speeding up Convolutional Neural Network Training

Frédéric Rayar<sup>(✉)</sup> and Seiichi Uchida

Kyushu University, Fukuoka 819-0395, Japan  
{rayar, uchida}@human.ait.kyushu-u.ac.jp

**Abstract.** We propose a fast hybrid statistical and graph-based sample preselection method for speeding up CNN training process. To do so, we process each class separately: some candidates are first extracted based on their distances to the class mean. Then, we structure all the candidates in a graph representation and use it to extract the final set of preselected samples. The proposed method is evaluated and discussed based on an image classification task, on three data sets that contain up to several hundred thousands of images.

**Keywords:** Convolutional neural network  
Training data set preselection · Relative Neighbourhood Graph

## 1 Introduction

Recently, Convolutional Neural Networks (CNN) [7] have achieved the state-of-the-art performances in many pattern recognition tasks. One of the properties of the CNN, that allows to achieve very good performance, is the multi-layered architecture (up to 152 layers for ResNet). Indeed, the additional hidden layers can allow to learn complex representation of the data, acting like an automatic feature extraction module. Another requirement to take advantage of CNN is to have at disposal large amounts of training data, that will be used to build a refined predictive model. By large amounts, we understand up to several millions labelled data, that will allow to avoid overfitting and enhance the generalisation performance of the model.

Nonetheless, the combination of deep neural networks and large amount of training data implies that substantial computing resources are required, for both training and evaluation steps. One of the solutions that can be considered is the hardware specialization, such as the usage of graphic processing units (GPU), field programmable gate arrays (FPGA) and application-specific integrated circuits (ASIC) like Google's tensor processing units (TPU). Another solution is sample preselection in the training data set. Indeed, several reasons can support the need of reducing the training set: (i) reducing the noise, (ii) reducing storage and memory requirement and (iii) reducing the computational requirement.

In a recent work [9], the relevance of a graph-based preselection technique has been studied and it has been experimentally shown that it allowed to reduce the training data set up to 76% without degrading the CNN recognition accuracy. However, one limitation of the proposed method was that the graph computation time could still be considered as high for large data sets. Hence, in this paper, we aim at addressing this issue and propose a fast sample preselection technique to speed up CNN training when using large data sets.

The contributions of this paper are as follows:

1. We propose a hybrid statistical and graph-based approach for preselecting training data. To do so, for each class, some candidates are first extracted based on their distances to the class mean. Then, we structure the candidates in a graph and use it to gather the final set of preselected samples.
2. We discuss the proposed preselection technique, based on experimentation on three data sets, namely CIFAR-10, MNIST and HW\_R-OID (50,000, 60,000 and 740,348 training images, respectively), in image classification tasks.

The rest of the paper is organised as follows: Sect. 2 presents the paradigms on sample preselection and briefly reminds the work that has been done previously in [9]. Section 3 presents the proposed hybrid statistical and graph-based preselection method. The experimentation details are given in Sect. 4 and the results that have been obtained are discussed in Sect. 5. Finally, we conclude this study in Sect. 6.

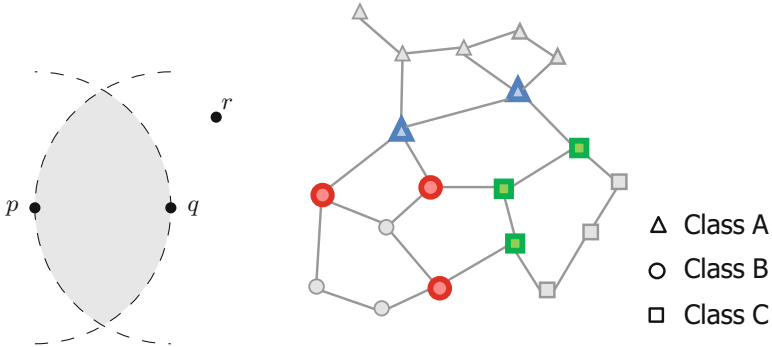
## 2 Related Work

### 2.1 Training Sample Selection

Several sample selection techniques have been proposed in the literature, to reduce the size of machine learning training data sets. They can be organised according to the following three paradigms:

1. “*editing*” techniques, that aim at eliminating erroneous instances and remove possible class overlapping. Hence, such algorithms behave as noise filters and retain class internal elements.
2. “*condensing*” techniques, that aim at finding instances that will allow to perform as well as a nearest neighbour classifier that uses the whole training set. However, as mentioned in [4], such techniques are “*very fragile in respect to noise and the order of presentation*”.
3. “*hybrid*” techniques (editing-condensing), that aim at removing noise and redundant instances at the same time.

These techniques exploit either: (i) random selection methods [8], (ii) clustering methods [15] or graph-based methods [12] to perform the sample selection. One can refer to thorough surveys that have been done recently: in 2012, Garcia et al. [2] focus on the sample selection for nearest neighbour based classification. Stratification technique is used to handle large data sets and no graph-based



**Fig. 1.** (Left) Relative neighbourhood (grey area) of two points  $p, q \in \mathbb{R}^2$ . If no other point lays in this neighbourhood, then  $p$  and  $q$  are relative neighbours. (Right) Illustration of bridge vectors on a toy data set. The bridges vectors are highlighted with colours and thicker borders. (Color figure online)

techniques has been evaluated. In 2014, Jung et al. [5] shed light on the sample preselection for Support Vector Machine (SVM) [1] based classification. However, they evaluated only post-pruning methods, to address issues of application engineers. As confirmed by the existence of the two aforementioned surveys, sample selection has been widely studied for the nearest neighbour classifier and the SVMs. However, to the best of our knowledge, no similar studies has been performed for CNN (or more generally neural networks). Conversely, the studies that use CNN usually focus on the acquirement of large training data sets, using crowdsourcing, synthetic data generation or data augmentation techniques.

## 2.2 Graph-Based Sample Selection

Toussaint et al. [12] have been the first in 1985 to study the usage of a proximity graph [13] to perform sample selection for nearest neighbour classifiers using Voronoi diagrams. Following this study, several other proximity graphs have been used to perform training data reduction such as: the  $\beta$ -skeleton, the Gabriel Graph (GG), and the Relative Neighbourhood Graph (RNG). In this last study, the authors conclude that the GG seems to be the best fit for sample selection. More recently Toussaint et al. have used a graph-based selection technique and in a comparison study [14] against random selection, they conclude that “*proximity graph is useless for speeding up SVM because of the computation times*” and assert that “*a naive random selection seems to be better*”. However, they only evaluated their work with a data set of 1641 instances.

In [9], the efficiency of using a condensing graph-based approach to select samples for training CNN on large data sets has been experimentally shown. To do so, the RNG, that has been proven a good fit to preselect high-dimensional samples [14] in large training data sets [3], has been used. The method consisted in: (i) building the RNG of the whole training data set and (ii) extracting

so-called “*bridge vectors*”, that correspond to nodes that are linked to another class node by an edge in the RNG. The bridge vectors are the final set of pre-selected training samples that are then fed to the CNN. Figure 1 illustrates the RNG relative neighbourhood definition (left) and the notion of bridge vectors (right). This preselected set allowed to reduce the training data set up to 76% without degrading the recognition accuracy, and performed better than random approaches. However, the RNG computation of the whole training data sets can remain an issue when dealing with large data sets. Hence, in this study, we aim at addressing this issue by proposing a fast hybrid statistical and graph-based preselection method.

### 3 Fast Hybrid Statistical and Graph-Based Sample Preselection

Since the issue of the RNG computation is related to the number of data in the whole training data set, one first idea that comes to mind is to take advantage of the supervised property of the CNN-based classification, and build an RNG for each class. Then, the preselection boils down to gather the data that lie in each class border. However, both exact (*e.g.* cluster boundaries) and approximative (*e.g.* low betweenness centrality nodes) approaches still require high computation requirements (*e.g.* all-pair shortest path computation). To address this, we propose to first extract some candidates for each class using a statistical approach, and then use a graph-based approach on the candidates subset.

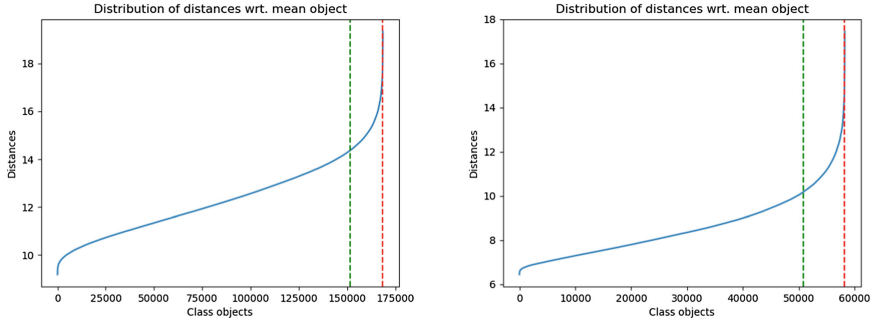
#### 3.1 Frontier Vectors

One of the goal of this study is to preselect samples that are similar to the bridge vectors presented in 2 (see Fig. 1 (right)). Since these bridge vectors may lie in the frontiers of classes, we propose to perform a simple statistical-based candidates selection for each class. To do so, for each class  $C$ , we: (i) compute the mean,  $\mu_C$ , (ii) compute the distances of each element  $x \in C$  to the mean,  $\delta(x, \mu_C)$ , (iii) sort these distances by ascending order, (iv) select elements that are above a given distance  $D$  to the mean.

The elements that are gathered in this way are among the farthest to the mean, hence they have a better chance to lie in the boundary of the class. The extracted candidates at this step are later called “*frontier vectors*” (FV). Figure 2 presents the plots of the sorted distance distribution of the two first classes of the HW\_R-OID data set.

#### 3.2 Automatic Threshold Computation

The last step to gather the frontier vectors of a given class, is to select elements that are above a given distance  $D$  to the mean. Given the shapes of the curves presented in Fig. 2, it corresponds to select the elements on the right part of the curve. The issue of the value of  $D$  arises: one naive solution could be to set



**Fig. 2.** Distribution of the sorted distances of a given class elements wrt. the class mean. We present here the distribution only for the two first classes of the largest data set (HW\_R-OID), due to space allowance. The red vertical dotted line corresponds to the threshold that is obtained using a basic maximum curvature criterion strategy, and the green one corresponds to one obtained the sliding-window maximum curvature criterion strategy. (Color figure online)

a value regarding the number of elements of the class. However, this strategy has two drawbacks: (i) it introduces an empirical parameter that may have an impact on the results and (ii) it does not fit the observations made during the study of [9] on the bridge vectors. Indeed, no direct relation was found between the number of elements of a class and its number of bridge vectors.

To address the automatic computation of this parameter, we propose to use a maximum curvature criterion. For a given data set, let us consider a given class  $C$ . We denote  $n$  the number of elements of  $C$ ,  $\mu$  the mean of  $C$ ,  $y$  the curve defined by the sorted distances  $\delta(x, \mu)$  of each element  $x \in C$  (in ascending order), and  $y'$ ,  $y''$  the first and second derivative of  $y$ , respectively. Then, we define the curvature criterion  $\gamma$  as follows:

$$\gamma(x) = \frac{y''}{(1 + y'^2)^{3/2}}, \text{ where } x \in \llbracket 1, n \rrbracket.$$

A naive strategy consists in finding the index of the maximum curvature value of  $y$ ; however, it may result in favouring indices associated to high values, and will gather only a few number of the class elements. This phenomenon could be seen in Fig. 2: the red vertical dotted lines correspond to the thresholds computed using the naive strategy.

To circumvent this problem, we propose to use a sliding window maximum curvature criterion strategy. Such a strategy has already been used efficiently in a previous work [10]. Let us define the set of windows  $W = \cup_{i \in \llbracket 1, n-m \rrbracket} W_i$ , where  $W_i = \{w_1^i, \dots, w_m^i\}$ .  $w_k^i \in \llbracket 1, n \rrbracket$  are the indices of window  $W_i$  and  $m$  is the size of

the windows. Hence, we have  $|W| = n - m + 1$  windows defined on the interval  $\llbracket 1, n \rrbracket$ . We then define the window’s curvature  $\gamma_i$ :

$$\gamma_i = \gamma(W_i) = \frac{\frac{1}{m} \sum_{w \in W_i} \gamma(w)}{\max_{w \in W_i} \gamma(w)}.$$

By selecting the maximum curvature over the set of windows, we have:

$$i^* = \operatorname{argmax}_{i \in \{1 \dots |W|\}} \gamma_i, \text{ and thus deduce } D = \delta(i^*, \mu).$$

Figure 2 illustrates the relevance of the sliding-window maximum curvature criterion strategy. We have set  $m = \frac{n}{10}$  to have a trade-off between the global and local maximum curvature. For a given data set and a given class, the green dotted vertical line in the plot corresponds to the value of  $i^*$  that has been automatically computed.

### 3.3 Overall Algorithm

Since the frontier vectors correspond to class boundaries, they may appear in a part of the feature space that do not correspond to classes frontiers. Hence, we use the bridge vectors extraction, proposed in the study of [9], but only on the frontier vector subset, addressing the high RNG computation time. Furthermore, this also allows to balance the fact that the proposed automatic threshold strategy does not extract only the farthest elements of a given class. The bridge vectors extracted at this step form the final preselected set of samples. We refer to these samples as “*frontier bridge vectors*” (FBV) in the rest of the paper. Algorithm 1 summarises the proposed hybrid statistical and graph-based sample preselection strategy.

## 4 Experimental Setup

### 4.1 Data Sets

To evaluate the proposed preselection method, we have used three data sets. First, the CIFAR-10 [6] data set is a subset of the Tiny Images [11] data set, that has been labelled. It consists of ten classes of objects with 6000 images in each class. The classes are: “airplane, automobile (but not truck or pickup truck), bird, cat, deer, dog, frog, horse, ship, and truck (but not pickup truck)”, as per the definition of the data set’s creator. We have used 50,000 images in the training data set and 10,000 for testing purpose. Second, the MNIST [7] data set, that corresponds to  $28 \times 28$  binary images of centered handwritten digits. Ground truth (*i.e.* correct class label (“0”, ..., “9”)), is provided for each image. In our experiments, we have used 60,000 images in the training data set and 10,000 for testing purpose. Last, the HW\_R-OID data set is an original data set from [16]. It contains 822,714 images collected from forms written by multiple people. The images are  $32 \times 32$  binary images of isolated digits and

---

**Algorithm 1.** Fast hybrid statistical and graph-based sample preselection algorithm

---

**Input:**  $DATA$  // data features per class  
**Input:**  $\delta$  // distance function  
**Output:**  $FBV$  // final preselected sample list

```

1  $FV = []$ 
2 for each class  $c$  do
3    $n =$  number of elements in  $c$ 
4    $m = \frac{n}{10}$ 
5   Compute class mean  $\mu$ 
6    $list = []$ 
7   for each  $x \in c$  do
8     Append  $\delta(x, \mu)$  to  $list$ 
9   end
10  Sort  $list$  (by ascending order)
11  Compute  $i^*$ 
12  Append elements of  $c$  at  $[[i^*, n]]$  to  $FV$ 
13 end
14  $RNG =$  Build graph from  $FV$ 
15  $FBV =$  Extract  $BV$  from  $RNG$ 

```

---

ground-truth is also available. In this data set, the number of the samples of each class is different but almost the same (between 65,000 and 85,000 samples per class, except the class “0” that has slightly more than 187,000 samples). In our experiments, we have split the data set in train/test subsets with a 90/10 ratio (740,438 training + 82,276 test images). To do so, 90% of each class samples have been gathered to build the training subset.

For the three aforementioned data sets, the intensities of the raw pixels have been used to describe the images, and the Euclidean distance has been used to compute the similarity between two images.

## 4.2 Workflow

The goal is to evaluate the relevance of the proposed preselection technique, but also compare its performance to the bridge vectors of the study of [9]. To do so, five different training subsets have been used for a given data set:

- WHOLE: the whole training data set,
- BV: only the extracted bridge vectors of the RNG build from WHOLE,
- FV: only the extracted frontier vectors of WHOLE,
- FBV: only the extracted bridge vectors of the RNG build from FV,
- $RANDOM_{FBV}$ : a random subset of WHOLE, with approximately the same size as FBV.



### 4.3 CNN Classification

Experiments were done on a computer with a i7-6850K CPU @3.60 GHz, with 64.0 GB of RAM (not all of it was used during runtime), and a NVIDIA GeForce GTX 1080 GPU. Our CNN classification implementation relies on the usage of Python (3.6.2), along with the Keras library (2.0.6) and a TensorFlow (1.3.0) backend.

The same CNN structure and parameters of the study of [9] have been used. Regarding the CNN architecture, namely modified LeNet-5 is used: the main difference with the original LeNet-5 [7] is the usage of ReLU and max-pooling functions for the two CONV layers. As mentioned in [16], it is “*a rather shallow CNN compared to the recent CNNs. However, it still performed with an almost perfect recognition accuracy*” (when trained with a large data set).

No pre-initialisation of the weights is done, and the CNN is trained with an Adadelta optimiser on 10 epochs for the two handwritten digit data sets, and an Adam optimiser on 100 epochs for the CIFAR-10 data set. The Adam optimiser has been chosen for the CIFAR-10 data set to avoid the strong oscillating behaviour during the training observed when using the Adadelta optimiser. During our experimentation, both computation times and recognition accuracies have been measured for further analysis. For each training data sets, experiments were run 5 times to compute an average value of the aforementioned metrics.

**Table 1.** *BV* and *FBV* preselection strategy computation times (in seconds).

	Data set	CIFAR-10	MNIST	HW_R-OID
BV	Data load	2	133	1,397
	RNG/BV computation	211	304	61,270
	Total	213	437	62,667
FBV	Data load	18	24	1,434
	Statistical pruning	3	4	147
	RNG/BV computation	9	5	622
	Total	40	32	2,203

## 5 Results

### 5.1 Preselection Method Computation Times and Data Reduction

One of the goal of the present study is to address the high RNG computation requirement observed during the preselection phase in large training data sets. Table 1 presents the computation times of the previous preselection strategy, namely the bridge vectors, and the one proposed in this study, namely the frontier bridge vectors. For the three data sets, a major speed-up ratio is obtained: 5.33, 13.65 and 28.44 for CIFAR-10, MNIST and HW\_R-OID, respectively.

For the largest data set, it represents a reduction of the preselection computation time from 17 h 25 m to 37 m.

Table 2 presents for each data sets, the size of the underlying training data sets in the first rows. Previously, using the bridge vectors as preselected samples, we have obtained a reduction of the training data set, up to 76%. By using the proposed hybrid preselection strategy, we achieve a data reduction that goes up to 96.57% (for the largest data set). Furthermore, we note that the hybrid approach, which extracts bridge vectors from the frontier vectors, allows its own data reduction. Indeed, this step allows to reduced the data, up to 69% between the *FV* and the *FBV*.

This reduction of the training data set has an expected impact on the CNN training time, with a speed-up ratio up to 15. The third rows of Table 2 present the average computation time per epoch.

**Table 2.** Classification results: (i) size of the training data set, (ii) average recognition accuracy and (iii) average training time per epoch (in seconds) are presented.

	Training data set	WHOLE	BV	FV	FBV	RANDOM <sub>FBV</sub>
CIFAR-10	# training data	50,000	41,221	8,713	6,845	6,850
	accuracy (%)	76.65	75.17	59.05	58.63	61.45
	epoch time (s)	42	35	9	7	7
MNIST	# training data	60,000	22,257	6,637	2,876	2,880
	accuracy (%)	98.79	98.78	96.22	95.25	94.69
	epoch time (s)	24	10	3	2	2
HW_R-OID	# training data	740,438	173,808	80,477	25,395	25,397
	accuracy (%)	99.9343	99.9314	99.7460	99.7085	99.4307
	epoch time (s)	412	107	56	27	27

## 5.2 Preselection Method Efficiency

Table 2 also presents the average accuracies obtained for all the training data sets introduced in Sect. 4.2 for the three data sets. Several observations can be made from these results.

For the two handwritten isolated digit data sets, we have:

$$\text{WHOLE} \approx \text{BV} > \text{FV} > \text{FBV} > \text{RANDOM}_{\text{FBV}} \quad (1)$$

Furthermore, the average recognition rates obtained using only the FBV are in the same order of magnitude to the ones obtained when using the whole training data set:  $-3.54\%$  and  $-0.2258\%$  for MNIST and HW\_R-OID, respectively. However, the same observation can be made for the RANDOM<sub>FBV</sub> training set, which may be interpreted as an indicator that either the data sets are lenient or that the FBV are not discriminative enough on their own in the training of the CNN.

For CIFAR-10, we observe a different behaviour than the one mentioned above. First, the relation described in Eq. 1 does not stand. Indeed, the average accuracy obtained for  $\text{RANDOM}_{\text{FBV}}$  is higher than both the ones of  $\text{FV}$  and  $\text{FBV}$ . Furthermore, the degradation in terms of average accuracy between  $\{\text{WHOLE}, \text{BV}\}$  and  $\{\text{FV}, \text{FBV}, \text{RANDOM}_{\text{FBV}}\}$  is no more negligible: around  $-16\%$ . These results may be due to the strong dissimilarity between this data set class elements.

## 6 Conclusion

In this paper, we have proposed a fast sample preselection method for speeding up convolutional neural networks training and evaluation. The method uses a hybrid statistical and graph-based approach to reduce the high computational requirement that was due to the graph computation. Hence, it allows to drastically reduce the training data set while having recognition rate of the same order of magnitude for two of the studied data sets.

Future works will be to perform experimentation on another data set, to evaluate the generalisation of the proposed method. We also aim at starting a formal study on the existence of “*support vectors*” for CNN.

**Acknowledgement.** This research was partially supported by MEXT-Japan (Grant No. 17H06100).

## References

1. Cortes, C., Vapnik, V.: Support-vector networks. *Mach. Learn.* **20**, 273–297 (1995)
2. Garcia, S., Derrac, J., Cano, J., Herrera, F.: Prototype selection for nearest neighbor classification: taxonomy and empirical study. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**, 417–435 (2012)
3. Goto, M., Ishida, R., Uchida, S.: Preselection of support vector candidates by relative neighborhood graph for large-scale character recognition. In: *ICDAR*, pp. 306–310 (2015)
4. Jankowski, N., Grochowski, M.: Comparison of instances selection algorithms I. Algorithms survey. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) *ICAISC 2004. LNCS (LNAI)*, vol. 3070, pp. 598–603. Springer, Heidelberg (2004). [https://doi.org/10.1007/978-3-540-24844-6\\_90](https://doi.org/10.1007/978-3-540-24844-6_90)
5. Jung, H.G., Kim, G.: Support vector number reduction: survey and experimental evaluations. *IEEE Trans. ITS* **15**, 463–476 (2014)
6. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Technical report, Computer Science Department, University of Toronto (2012)
7. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**, 2278–2324 (1998)
8. Lee, Y.J., Huang, S.Y.: Reduced support vector machines: a statistical theory. *IEEE Trans. Neural Netw.* **18**, 1–13 (2007)
9. Rayar, F., Goto, M., Uchida, S.: CNN training with graph-based sample preselection: application to handwritten character recognition. *CoRR* abs/1712.02122 (2017)

10. Razafindramanana, O., Rayar, F., Venturini, G.: Alpha\*-approximated delaunay triangulation based descriptors for handwritten character recognition. In: ICDAR, pp. 440–444 (2013)
11. Torralba, A., Fergus, R., Freeman, W.T.: 80 million tiny images: a large data set for nonparametric object and scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **30**, 1958–1970 (2008)
12. Toussaint, G.T., Bhattacharya, B.K., Poulsen, R.S.: The application of Voronoi diagrams to non-parametric decision rules. *Comput. Sci. Stat.* 97–108 (1985)
13. Toussaint, G.T.: Some unsolved problems on proximity graphs (1991)
14. Toussaint, G.T., Berzan, C.: Proximity-graph instance-based learning, support vector machines, and high dimensionality: an empirical comparison. In: Perner, P. (ed.) *MLDM 2012. LNCS (LNAI)*, vol. 7376, pp. 222–236. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-31537-4\\_18](https://doi.org/10.1007/978-3-642-31537-4_18)
15. Tran, Q.A., Zhang, Q.L., Li, X.: Reduce the number of support vectors by using clustering techniques. In: *ICMLC*, pp. 1245–1248 (2003)
16. Uchida, S., Ide, S., Iwana, B.K., Zhu, A.: A further step to perfect accuracy by training CNN with larger data. In: *ICFHR*, pp. 405–410 (2016)



# UAV First View Landmark Localization via Deep Reinforcement Learning

Xinran Wang, Peng Ren<sup>(✉)</sup>, Leijian Yu, Lirong Han, and Xiaogang Deng

College of Information and Control Engineering,  
China University of Petroleum (East China), Qingdao 266580, China  
wangxinranupc@163.com, yuleijianupc@126.com, lironghan\_upc@163.com,  
{pengren, dengxiaogang}@upc.edu.cn

**Abstract.** In recent years, the study of Unmanned Aerial Vehicle (UAV) autonomous landing has been a hot research topic. Aiming at UAV's landmark localization, the computer vision algorithms have excellent performance. In the computer vision research field, the deep learning methods are widely employed in object detection and localization. However, these methods rely heavily on the size and quality of the training datasets. In this paper, we propose to exploit the Landmark-Localization Network (LLNet) to solve the UAV landmark localization problem in terms of a deep reinforcement learning strategy with small-sized training datasets. The LLNet learns how to transform the bounding box into the correct position through a sequence of actions. To train a robust landmark localization model, we combine the policy gradient method in deep reinforcement learning algorithm and the supervised learning algorithm together in the training stage. The experimental results show that the LLNet is able to locate the landmark precisely.

**Keywords:** Deep reinforcement learning · UAV  
Landmark localization

## 1 Introduction

The Unmanned Aerial Vehicles (UAVs) have many advantages such as low costs, easy-to-control flight routes and have the ability to automatically complete complex tasks. The combination of UAV and computer vision has extensive applications in many fields such as public safety, post-disaster rescue, information collection, video surveillance, transportation management and video shooting [1]. With the continuous development of UAVs, how to land successfully is an important part in UAV's applications. During the UAV's landing procedure, the landmark localization is the first step, which tells the UAV where to land. The landmark incorrect localization and the low accuracy of landmark localization are the main reasons that lead to UAV's landing failure [2]. Therefore, it is of great value to study the landmark localization of UAVs.

In recent years, the problem of locating object in videos has been studied by many researchers, which aims to identify the target object with a bounding box [3, 4]. To solve this problem, using convolution neural networks (CNNs) has attracted a lot of attention [5–7]. Further more, these methods like the R-CNN proposed by Girshick et al. [8, 9] have been proved to have effective performance [10, 11]. However, due to the difficulties in identification and localization problems, CNN models [5–7, 12, 13] require to be trained through a large amount of labeled training sequences [14]. However, there is no existing training datasets in the UAV landing scenarios. In contrast, reinforcement learning methods need relatively less data to train the model.

Reinforcement learning is an important research topic in machine learning. It does not require training based on samples, but interacts with the external environment, and receives environmental feedbacks and evaluation results to select the next action at the next time step. Reinforcement learning is inspired by the organism’s ability which interacts with the environment through trial and error mechanisms and learns the optimal strategy by maximizing the sum of reward [15].

Markov Decision Process (MDP) is a fundamental method in reinforcement learning. This mathematical frame provides a solution for decision making problems whose outcomes are partially random and partially under the control of the decision maker. An MDP has five elements, including a finite set of states  $S$ , a finite set of actions  $A$ , the state transition probability  $P_{sa}$ , the reward function  $R_a$  and the discount factor  $\gamma$ . The agent chooses an action according to the current state, interacts with the environment, observes the next action and gets a reward. The target of reinforcement learning is to get an optimal policy for a specific problem, such that the reward obtained under this strategy is the largest [15].

Deep reinforcement learning combines the perception of deep learning with the decision-making ability of reinforcement learning. It has the ability to control agents directly based on input, achieve end-to-end learning, directly learn and control strategies from high dimensional raw data. Deep reinforcement learning is an altricial intelligence method that closing to human thinking. The DeepMind group was among the first to conduct deep reinforcement learning research [16]. Then, DeepMind further developed an improved version of Deep Q Network [17], which has attracted widespread attention. Deep reinforcement learning is able to use perceptual information such as vision as input, and then output actions directly through deep neural networks without hand-crafted features. Deep reinforcement learning has the potential to enable agents to fully autonomously learn one or more skills like human.

Deep Q Network and policy gradient are two popular methods in deep reinforcement learning algorithms. The main method of the Deep Q Network algorithm is experience replay, which stores the data obtained from the exploration of the environment and then randomly sampling the samples to update the parameters of the deep neural network. Policy gradient method directly optimizes a parameterized control policy by a variant of gradient descent [18]. Unlike value

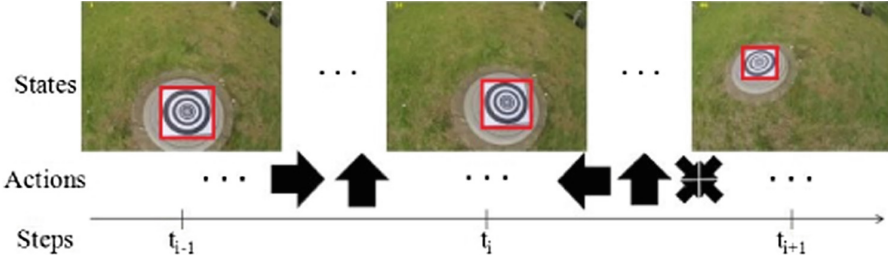


Fig. 1. State changes by taking a sequence of actions.

function approximation approach that gets policies from estimated value functions indirectly, the policy gradient method maximize the expected return of the policy. In our model we use the policy gradient method in the reinforcement learning training stage. In our work, to deal with the problem of landmark localization, we propose an effective method which is inspired by deep reinforcement learning. Our method is achieved by transforming the bounding box through a sequence of actions, making the box coincidence with the landmark. In Fig. 1, we illustrate the steps of the network’s decision process about how to locate the landmark.

## 2 Landmark Localization as an Action Dynamic Process

To solve the landmark localization problem, we exploit the LLNet, which controls the sequential actions to locate the target. We describe the architecture of the LLNet in Fig. 2. To initialize our network, we use a small CNN, the pre-trained VGG-M [19]. As shown in Fig. 2, the LLNet that we proposed has three convolutional layers. {fc4, fc5} are the next two fully connected layers. The output of the CNN is concatenated with the action history vector  $h_t$ . The {fc6, fc7} layer predict the action probability and the confidence score.

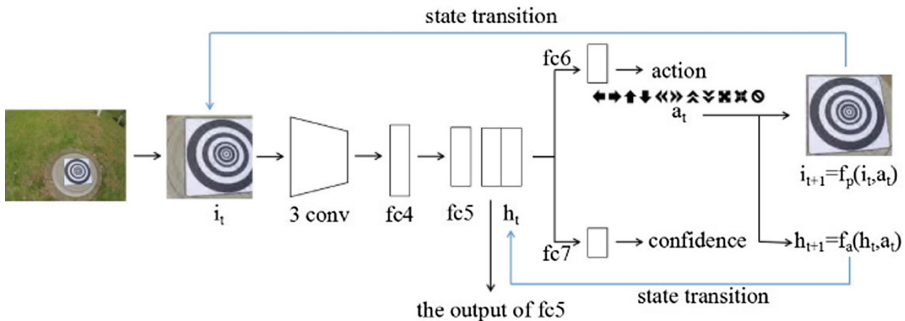


Fig. 2. Architecture of the proposed LLNet.

The LLNet is trained by both supervised learning and reinforcement learning. Training with supervised learning, the LLNet learns how to locate the landmark when there is no sequential information. The trained network from the supervised learning training stage is used as the initial network for the reinforcement learning training stage. We use the policy gradient method in reinforcement learning to train action dynamics of the landmark.

## 2.1 Proposed Approach

To achieve the landmark localization process, we follow the MDP method. In our landmark localization model, we describe the MDP as a process that the goal of the agent is to locate the landmark with a bounding box. We consider a single image as the environment. The way how the agent transforms the bounding box follows a set of actions. For each image, the agent generates a sequence of actions until it finally locates the landmark. The agent receives positive and negative rewards at the last state of the image, the value of the reward is decided by whether the agent locates the landmark successfully. Specifically, we follow the deep reinforcement learning scheme [14] to construct our framework.

**Action:** The set of actions  $A$  is defined as an eleven dimensional vector as shown in Fig. 3. Specifically, the actions include four vertical and horizontal actions {left, right, up, down}, their two times larger moves, scale changing actions {bigger, smaller} and the trigger action to stop the locating process. In this way, the localization box is able to transform in four degrees of freedom.



**Fig. 3.** The definition of the set of actions  $A$ .

**State:** We describe the state  $s_t$  as a tuple  $(i_t, h_t)$ .  $i_t$  represents the image block in the localization box.  $h_t \in R^{110}$  is a binary vector contains the past 10 actions, whose values are set to be zero except the one takes action.  $b_t$  is a 4-dimensional vector and  $b_t = [x^{(t)}, y^{(t)}, w^{(t)}, l^{(t)}]$ , where  $(x^{(t)}, y^{(t)})$  represents the center position of the box,  $w^{(t)}$  is the width of the bounding box and  $l^{(t)}$  is the length of the box. In each image  $I$ , the  $i_t$  is described as:

$$i_t = \phi(b_t, I) \quad (1)$$

**State Transition Function:** The state transition function includes two parts: landmark transition function  $f_l(\cdot)$  and action dynamic function  $f_a(\cdot)$ . The box



transition function is described as  $b_{t+1} = f_l(b_t, a_t)$ . The change of the bounding box is described as:

$$\Delta x^{(t)} = \alpha w^{(t)} \text{ and } \Delta y^{(t)} = \alpha l^{(t)} \quad (2)$$

in our experiments, we set  $\alpha$  to be 0.03.

The action dynamic function  $f_a(\cdot)$  is described through the action history vector  $h_t : h_{t+1} = f_a(h_t, a_t)$ .

**Reward Function:** To improve the performance of the agent of locating the landmark, the reward function is defined as  $R$ . It describes the reward that the agent receives when it takes action  $a$  to move to state  $s_{t+1}$  from state  $s_t$ . In our framework, we use Intersection-over-Union (IoU) between the located landmark and the bounding box in every image to measure the performance of the model.  $\text{IoU}(b, g) = \text{area}(b \cap g) / \text{area}(b \cup g)$ .

We use  $b$  to represent the located target region and  $g$  to represent the ground truth box of the target object. The reward function is defined as follows:

$$R(s_t) = \text{sign}(\text{IoU}(b', g) - \text{IoU}(b, g)) \quad (3)$$

The reward is positive when the IoU improves from state  $s_t$  to state  $s_{t+1}$ , and negative otherwise. The reward function suits any action to transform the box. When there are no other actions in transforming the bounding box, the agent then achieves the final step  $T$  and should choose the trigger action. The trigger action does not change the bounding box, and the IoU is zero at the final step. Thus, as for the trigger action, the reward function is assigned by

$$R(s_T) = \begin{cases} \eta, & \text{if } \text{IoU}(b_T, g) > \tau \\ -\eta, & \text{otherwise} \end{cases} \quad (4)$$

where  $\eta$  is the reward for the trigger action, and  $\tau$  represents the minimum IoU allowed. In our experiments, we set  $\eta$  as 1 and  $\tau$  as 0.7 during the training process.

### 3 LLNet’s Training

In this section, we explain how to train the LLNet with both supervised learning and reinforcement learning. In the supervised learning stage, the LLNet predicts an action according to the current state. In the reinforcement learning stage, we use the pre-trained network from the supervised learning stage as the initial network and the LLNet is trained by using the policy gradient algorithm [20].

#### 3.1 Supervised Learning Training

While training with the supervised learning, the training image samples including three parts: image blocks  $i_j$ , action labels  $l_j^{(\text{act})}$  and class labels  $l_j^{(\text{cls})}$ .

The action dynamic is not taken into consideration in this part of training. We describe the ground truth box as  $g$ . For each training sample image block, the corresponding action label is defined as follows:

$$l_j^{(\text{act})} = \arg \max_a \text{IoU}(\bar{f}(i_j, a), g) \quad (5)$$

where  $\bar{f}(i_j, a)$  represents the changed box of  $i_j$  after taking action  $a$ . The class labels  $l_j^{(\text{cls})}$  is defined as follows:

$$l_j^{(\text{cls})} = \begin{cases} 1, & \text{if IoU}(i_j, g) > \tau \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

The training batch includes training samples  $\left\{ \left( i_j, l_j^{(\text{act})}, l_j^{(\text{cls})} \right) \right\}_{j=1}^n$ . The samples are formed by random selection. We train the LLNet by minimizing the multi-task loss function, defined as:

$$L_{SL} = \frac{1}{n} \sum_{j=1}^n L(l_j^{(\text{act})}, \hat{l}_j^{(\text{act})}) + \frac{1}{n} \sum_{j=1}^n L(l_j^{(\text{cls})}, \hat{l}_j^{(\text{cls})}) \quad (7)$$

where  $n$  represents the batch size,  $L$  represents the cross-entropy loss, and the predicted action and class is represented by  $\hat{l}_j^{(\text{act})}$  and  $\hat{l}_j^{(\text{cls})}$ , respectively.

### 3.2 Reinforcement Learning Training

While training with reinforcement learning, we train the network parameters  $N_{RL}(n_1, \dots, n_6)$ , except the fc7 layer, which is needed in locating phase. The purpose of reinforcement learning is to learn the state-action policy. At this training stage, the LLNet uses the training sequence and action dynamics to perform the simulation. At each iteration, the action history vector  $h_t$  is updated. In the training process, the training sequences  $\{I_l\}_{l=1}^m$  and the ground truth  $\{g_l\}_{l=1}^m$  are chosen randomly. In the simulation, the network produces a set of states  $\{s_{t,l}\}$ , actions  $\{a_{t,l}\}$  and rewards  $\{R(s_{t,l})\}$ ,  $l = 1, 2, \dots, m$  at the steps  $t = 1, 2, \dots, T_l$ . At the state  $s_{t,l}$ , the action  $a_{t,l}$  is defined as:

$$a_{t,l} = \arg \max_a p(a|s_{t,l}; N_{RL}) \quad (8)$$

where  $N_{RL}$  represents the initial reinforcement learning network,  $p(a|s_{t,l})$  represents the action probability.

When the simulation is finished, the scores of the localization  $\{v_{t,l}\}$  are calculated with the ground truth  $\{g_l\}$ . In the final state, the localization score is  $v_{t,l} = R(s_{T_l,l})$ . More specifically, the score increases by 1 if the localization is successful. Otherwise, the score reduces by 1. To maximize the localization scores, the  $N_{RL}$  complies with the following condition:

$$\Delta N_{RL} \propto \sum_l^L \sum_t^{T_l} \frac{\partial \log p(a_{t,l}|s_{t,l}; N_{RL})}{\partial N_{RL}} v_{t,l} \quad (9)$$

Even if the ground truth is partially known, our framework is still able to train the LLNet successfully. While training the LLNet with reinforcement learning, the localization scores  $\{v_{t,l}\}$  should be determined. However, in the unlabeled sequences, it is unable to determine the localization scores. To solve this problem, we assign the localization scores to the reward obtained from the result of the simulation.

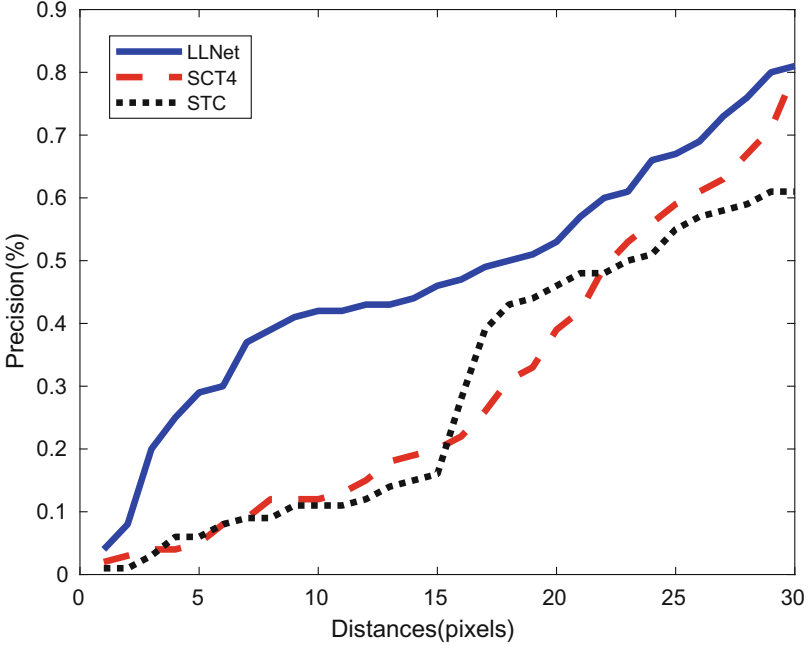
## 4 Experiments

In the experiments, we use the captured video with the UAV’s downward looking camera to train and validate the proposed LLNet. For the training datasets, the video frames are annotated with the coordinates of the corner of the landmark. To make a robust landmark localization policy, we use the VOT2015 [21] and 300 captured video frames to train the LLNet. We evaluate the LLNet on other 500 unannounced video frames. The first frame is distortionless, and the landmark can be localized by the edge detection methods. After that, the LLNet will locate the landmark through deep reinforcement learning.



**Fig. 4.** UAV landmark localization results from different heights and rotations.

The results of the experiment are shown in Fig. 4. The LLNet is able to localize the landmark in all testing frames. It means that our LLNet method can locate the landmark robustly with different heights and rotations. Furthermore,



**Fig. 5.** Percentage of frames with respect to the pixel distance between located center position and the ground truth.

to verify the effectiveness of the LLNet, we compare the performance of LLNet with other two methods. In Fig. 5 we show the percentage of frames with respect to the pixel distance of the located center position with that of the ground truth. For the evaluation, we include the STC [22] and the SCT4 [23]. The results indicate that the center position located by the LLNet is precise. Focus on the distance between the located position and the ground truth at the range of 0 to 30 pixels, the LLNet has higher precision than the STC and the SCT4 at all time. In the experiment of the LLNet there is no more than 30 error pixels in over 80% testing frames while the percentage of the STC method is only 60%. The comparison results show that our method achieved the better performance compared to other methods.

## 5 Conclusion

In this paper, we have proposed the LLNet to solve UAV landmark localization problems. The proposed approach is typically different from other object localization method. Through our work, reinforcement learning is an efficient algorithm for object localization problems. The agent is able to learn from its own history mistakes and find the best policy to locate the landmark position precisely.

## References

1. Luo, C., Yu, L., Ren, P.: A vision-aided approach to perching a bio-inspired unmanned aerial vehicle. *IEEE Trans. Ind. Electron.* **65**(5), 3976–3984 (2018)
2. Yu, L., et al.: Deep learning for vision-based micro aerial vehicle autonomous landing. *Int. J. Micro Air Veh.* (2018)
3. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The Pascal Visual Object Classes (VOC) challenge. *Int. J. Comput. Vis.* **88**(2), 303–338 (2010)
4. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **115**(3), 211–252 (2015)
5. Hong, S., You, T., Kwak, S., Han, B.: Online tracking by learning discriminative saliency map with convolutional neural network. In: *International Conference on Machine Learning*, pp. 597–606 (2015)
6. Nam, H., Han, B.: Learning multi-domain convolutional neural networks for visual tracking. In: *Computer Vision and Pattern Recognition*, pp. 4293–4302 (2016)
7. Wang, N., Li, S., Gupta, A., Yeung, D.-Y.: Transferring rich feature hierarchies for robust visual tracking (2015)
8. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Computer Vision and Pattern Recognition*, pp. 580–587 (2014)
9. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: *Advances in Neural Information Processing Systems*, pp. 91–99 (2015)
10. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: integrated recognition, localization and detection using convolutional networks (2013)
11. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014)
12. Li, H., Li, Y., Porikli, F.: Robust online visual tracking with a single convolutional neural network. In: Cremers, D., Reid, I., Saito, H., Yang, M.H. (eds.) *ACCV 2014*. LNCS, vol. 9007, pp. 194–209. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-16814-2\\_13](https://doi.org/10.1007/978-3-319-16814-2_13)
13. Wang, L., Ouyang, W., Wang, X., Lu, H.: Visual tracking with fully convolutional networks. In: *International Conference on Computer Vision*, pp. 3119–3127 (2015)
14. Yun, S., Choi, J., Yoo, Y., Yun, K., Choi, J.Y.: Action-decision networks for visual tracking with deep reinforcement learning (2017)
15. Sutton, R.S., Barto, A.G.: *Introduction to Reinforcement Learning*. MIT Press, Cambridge (1998)
16. Silver, D., et al.: Mastering the game of go with deep neural networks and tree search. *Nature* **529**(7587), 484–489 (2016)
17. Mnih, V., et al.: Human-level control through deep reinforcement learning. *Nature* **518**, 529–533 (2015)
18. Sammut, C., Webb, G.I.: *Encyclopedia of Machine Learning And Data Mining*. Springer, Boston (2017)
19. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: delving deep into convolutional nets (2014)
20. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. In: Sutton, R.S. (ed.) *Reinforcement Learning*, pp. 5–32. Springer, Boston (1992). [https://doi.org/10.1007/978-1-4615-3618-5\\_2](https://doi.org/10.1007/978-1-4615-3618-5_2)

21. Kristan, M., et al.: The visual object tracking VOT2015 challenge results. In: International Conference on Computer Vision Workshops, pp. 1–23 (2015)
22. Zhang, K., Zhang, L., Liu, Q., Zhang, D., Yang, M.-H.: Fast visual tracking via dense spatio-temporal context learning. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 127–141. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-10602-1\\_9](https://doi.org/10.1007/978-3-319-10602-1_9)
23. Choi, J., Chang, H.J., Jeong, J., et al.: Visual tracking using attention-modulated disintegration and integration. In: Computer Vision and Pattern Recognition, pp. 4321–4330 (2016)



# Context Free Band Reduction Using a Convolutional Neural Network

Ran Wei<sup>1</sup>, Antonio Robles-Kelly<sup>1,2(✉)</sup>, and José Álvarez<sup>1</sup>

<sup>1</sup> DATA61 - CSIRO, Black Mountain Laboratories,  
Acton ACT 2601, Canberra, Australia  
[antonio.robles-kelly@data61.csiro.au](mailto:antonio.robles-kelly@data61.csiro.au)

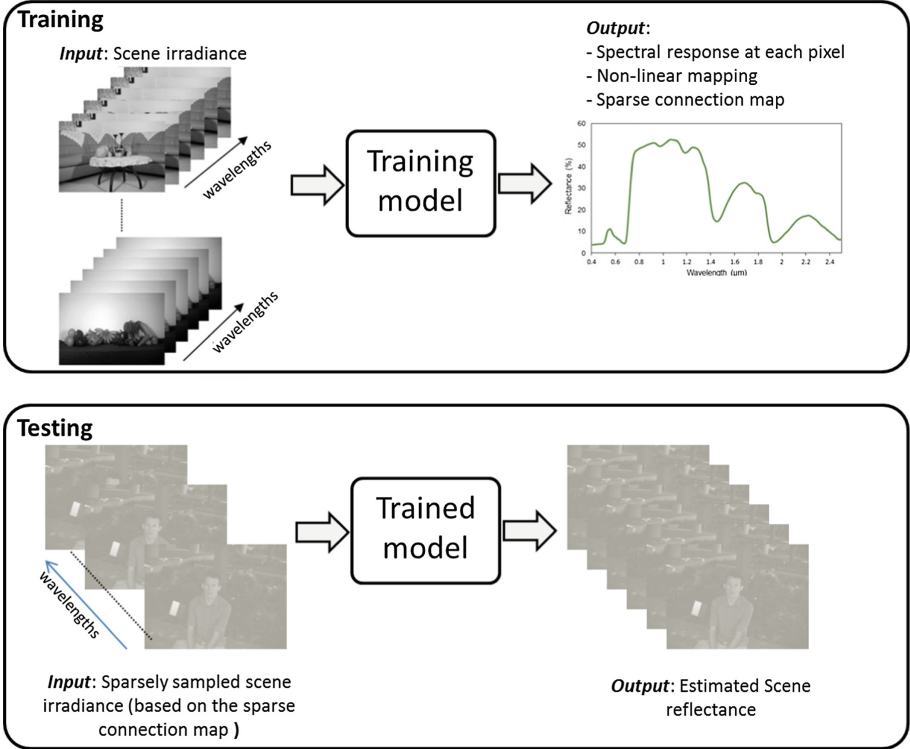
<sup>2</sup> School of Information Technology, Deakin University,  
Waurin Ponds, VIC 3216, Australia

**Abstract.** In this paper, we present a method for content-free band selection and reduction for hyperspectral imaging. Here, we reconstruct the spectral image irradiance in the wild making use of a reduced set of wavelength-indexed bands at input. To this end, we use of a deep neural net which employs a learnt sparse input connection map to select relevant bands at input. Thus, the network can be viewed as learning a non-linear, locally supported generic transformation between a subset of input bands at a pixel neighbourhood and the scene irradiance of the central pixel at output. To obtain the sparse connection map we employ a variant of the Levenberg-Marquardt algorithm (LMA) on manifolds which is devoid of the damping factor often used in LMA approaches. We show results on band selection and illustrate the utility of the connection map recovered by our approach for spectral reconstruction using a number of alternatives on widely available datasets.

## 1 Introduction

Compared to traditional monochrome and trichromatic cameras, hyperspectral image sensors can provide an information-rich representation of the spectral response of materials which poses great opportunities and challenges on material identification [4]. Furthermore, imaging spectroscopy enables the capture of the scene irradiance so as to recover the spectral reflectance and illuminant power spectrum for applications such as material-specific colour rendition [7], accurate colour reproduction [19] and material reflectance substitution [8]. Furthermore, the accurate reproduction and capture of the scene colour across different devices is an important and active area of research spanning color correction [6], camera simulation [13], sensor design [5] and white balancing [11].

Note that hyperspectral imaging technologies can capture image data in tens or hundreds of bands covering a broad spectral range. As a result, band reduction or selection on the spectral image data has been used in order to reduce its dimensionality for tasks such as unmixing [22], super-resolution [1] and material classification [9]. Here we note that, band selection is eminently task driven,

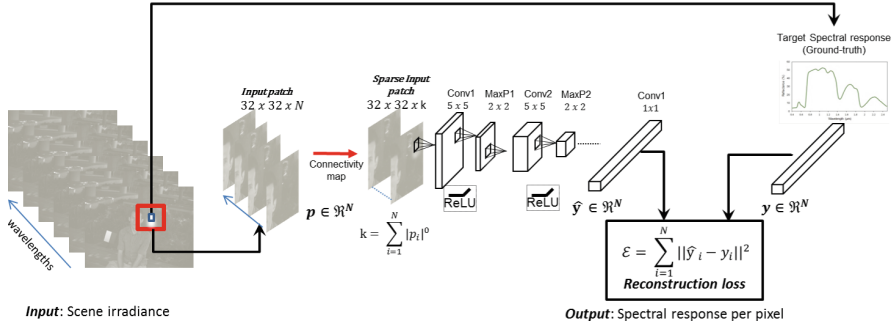


**Fig. 1.** Our approach aims at learning a generic mapping between a subset of wavelength-indexed bands and the scene irradiance. At training, we use spectral images to learn a sparse input connection map and a locally supported, non-linear generic transformation between the subset of wavelength-indexed bands at a pixel neighbourhood and its actual spectrum. At testing, the subset of spectral bands are used to reconstruct the full spectral irradiance.

whereby the task in hand determines the bands to be selected for further consideration. In the other hand, band reduction often aims at preserving the information in the spectral image for encoding and compression [3]. Moreover, band selection is often aimed at removing the redundancy in the image data so as to reduce the computational burden for encoding, classification and interpretation tasks whereas dimensionality reduction approaches are often used so as to obtain a lower-dimensional representation of the image. As a result, these methods often lack the generality for “content-free” band selection aimed at reconstructing the image irradiance “in the wild”. This is a major advantage of our algorithm, which can perform band reduction independently of the image contents.

The work presented here is somewhat related to spectral reconstruction in the sense that we seek to recover the spectral irradiance from a reduced set of wavelength indexed bands. Here, however, we aim a developing a “content free” approach that does not depend upon the application in hand or the sensitivity





**Fig. 2.** Proposed framework for learning a spectral reconstruction mapping using only a reduced set of input bands.

function of a particular trichromatic camera or rendering content. This is important since, even when the camera has been radiometrically calibrated, the image raw colour values are sensor specific [15]. For instance, in [16] the authors propose an approach to reconstruct the scene’s spectral irradiance by learning a mapping between spectral responses and their RGB values for a given make and model of a camera. In [18], the author employs sparse coding and texture features to reconstruct the image irradiance assuming the sensitivity functions of the camera used to acquire the RGB input image are known.

Here we employ a convolutional neural network which, by using a connection table, can learn an input mapping. In this manner, we learn a generic non-linear transformation between a subset of wavelength indexed bands and the scene irradiance such that, once trained, our deep network can be used to obtain scene irradiance spectra making use of a much reduced set of wavelength indexed bands, *i.e.* channels, with a comparable spectral resolution to that of much more complex hyperspectral cameras.

To the best of our knowledge, there are no similar learning based approaches aiming to find the relevant input feature maps for band selection. However, methods such as DropConnect do aim at regularising large fully connected layers where a set of randomly selected weights is set to zero. In [2], sparse constraints are used for regularising the training process of a deep neural network. Also, it is worth noting in passing that although connection maps are not currently used, they were originally introduced in [12] to reduce the number of parameters and, hence, the complexity of deep networks. In [12], however, the connection map is a binary one which is used to “disconnect” a random set of feature maps. These contrast with our method, which aims at recovering a sparse input connection map with non-binary weights. To some extent, this architecture can be related to a dropout layer [20]. However, in dropout layers each feature detector is deleted randomly with a predefined probability and mainly aimed at regularising the network by removing certain units and back-propagates through the others.

## 2 Content-Free Band Selection

In this section we present our approach to learn a generic non-linear transformation between a subset of wavelength indexed bands and the scene irradiance. Our approach not only learns the mapping to recover the spectral response of every pixel in an image but also the optimal subset of bands (input channels) to perform the reconstruction. Contrary to other methods, our approach is content-free. That is, a method that does not depend on the application (contents of the scene) or the camera being used for acquiring the images. As shown in Fig. 1, the outcome is a model that, given a multispectral camera providing the subset of wavelengths, can yield scene irradiance spectra that is in close accordance with that captured by much more complex hyperspectral cameras. A straightforward application of our algorithm is reducing the cost for obtaining hyperspectral images while using acquisition sensors with lower number of bands.

### 2.1 Network Architecture

Our approach is based on the end-to-end architecture shown in Fig. 2 for simultaneously learning the parameters to recover the spectra response and optimising the number of input wavelengths required. Intuitively, we need a procedure that can disconnect an input component if its contribution is not relevant. In our particular case, we target disconnecting information provided by an input wavelength (image channel). To this end, our model introduces a connectivity map to define whether an input channel is relevant to the process or, in the contrary, it can be completely removed.

Consider a convolutional layer with convolutional kernel weights  $W \in \mathbb{R}^{m \times n \times d \times d}$  and bias  $\mathbf{b} \in \mathbb{R}^m$ , where  $n$  is the number of input channels (bands),  $m$  is the number of outputs and  $d$  represents the size of the convolutional kernel. The output of the  $i$ -th neuron  $z_i$  is related to the input data  $X$  according to,

$$z_i = \sigma\left(\sum_j (W_{ij}X_j + b_i)\right), \quad (1)$$

where  $\sigma$  is the activation function which is set to ReLU in our experiments  $\sigma(x) = \max(0, x)$ .

Our goal is to learn a subset of input channels to recover with high precision the spectral response of a camera. That is, we aim at reducing the redundancy existing between input channels and estimating which of them are necessary to recover the complete spectral response. To this end, we introduce a connectivity map  $p$  to control the influence of each input channel:

$$z_i = \sigma\left(\sum_j p_j(W_{ij}X_j + b_i)\right), \quad (2)$$

where  $p_j$  defines the connectivity of the  $j$ -th input channel to the network. Therefore, setting  $p_i$  to zero that particular feature map is made redundant and thus, does not contribute to any of the output feature maps. Note that

our formulation relaxes the binary constraint placed on selecting the number of input planes. The entries of our input connectivity map are trainable and can adopt any real number  $p_i \in [0 \dots 1]$  and thus, defining the relevance of the  $j$ -th input channel to the reconstruction of the spectral response.

Our network architecture consists of five convolutional layers followed by rectifier linear units after every convolution and pooling layers after the first three convolutional layers. Specific details of the network are shown in Fig. 2. The output of the network is a  $N$ -dimensional feature vector representing the spectral response of the central pixel of the input patch. The loss is computed as the mean squared error between the raw output and the spectral response obtained during the acquisition process.

The parameters of the network and the connectivity map are learned jointly using an alternating method. First, we fix the connectivity map and learn the parameters of the network using stochastic gradient descent with momentum. The loss for training the model is the minimum squared error between the output of the network, that is, the estimation of the spectral response of the target pixel and the spectral response of the same target pixel as acquired by the camera. Then, given a set of parameters for the network, we optimise the connectivity map enforcing its sparsity using the Levenberg-Marquardt algorithm. We train the network from scratch and the connection map is initialized to 1. That is, at the beginning of the process, all input channels are considered.

## 2.2 Sparse Connection Map Computation

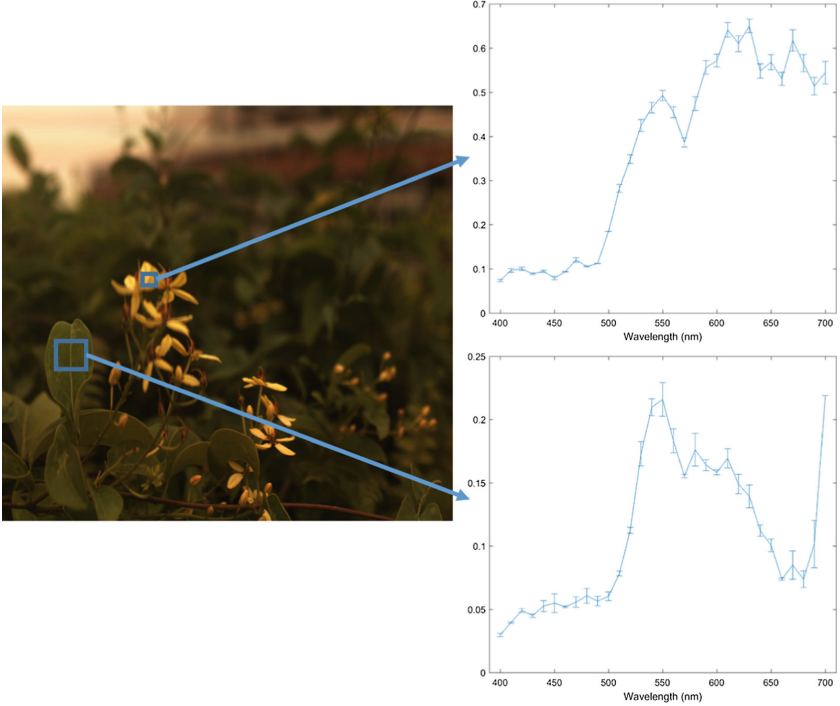
Now, we turn our attention to the computation of a sparse connection map  $\mathbf{p}$ . To this end, we aim at solving the optimization problem given by

$$\begin{aligned} \min_{\mathbf{p}} \quad & \left\{ \epsilon + \lambda |\mathbf{p}|_1 \right\} \\ \text{s.t.} \quad & p_i^2 \leq \tau \quad \forall p_i \in \mathbf{p} \\ & p_i \geq 0 \quad \forall p_i \in \mathbf{p} \end{aligned} \tag{3}$$

where  $\epsilon$  is the reconstruction error for the current state of the net,  $|\cdot|_p$  denotes the  $p$ -norm and  $\lambda$  is a scalar that accounts for the contribution of the second term in Eq. 3 to the minimization in hand. Note that, in the equation above, we have imposed a positivity constrain on  $p_i$  and defined  $\tau$  as a bounding positive constant which, in all our experiments, we have set to unity.

For the minimisation of the target function we have used a variant of the Riemannian Levenberg-Marquardt approach presented in [23]. The Levenberg-Marquardt Algorithm (LMA) [14] is an iterative trust region procedure [17] which provides a numerical solution to the problem of minimising a function over a space of parameters. For purposes of minimising the cost function, we commence by writing the cost function above in terms of the connection map entries.

Thus, at each iteration of the optimisation process, the new estimate of the parameter set is defined as  $\mathbf{p} + \delta$ , where  $\delta$  is an increment in the parameter space



**Fig. 3.** Spectral irradiance plots for two sample regions on an testing image from the NUS dataset. In the plots, the trace accounts for the mean spectral irradiance whereas the error-bars represent the variance of the spectral difference for the corresponding spectra yielded by our net trained using the Scyllarus dataset imagery with a  $\lambda = 0.03$ .

and  $\mathbf{p}$  is the current estimate of the transformation parameters. To determine the value of  $\delta$ , let  $g(\mathbf{p}) = \sqrt{\epsilon + \lambda|\mathbf{p}|_1}$  be the posterior probability evaluated at iteration  $t$  approximated using a Taylor series such that

$$g(\mathbf{p} + \delta) \approx \sqrt{\epsilon + \lambda|\mathbf{p}|_1} + \mathcal{J}\delta \quad (4)$$

where  $\mathcal{J}$  is the Jacobian of  $\frac{\partial g(\mathbf{p}+\delta)}{\partial \mathbf{p}}$ .

The set of equations that need to be solved for  $\delta$  is obtained by equating to zero the derivative with respect to  $\delta$  of the equation resulting from substituting Eq. 4 into the cost function. Let the matrix  $\mathbf{J}$  be comprised by the entries  $\frac{\partial g(\mathbf{p}+\delta)}{\partial \mathbf{p}}$ , *i.e.* the element indexed  $j, k$  of the matrix  $\mathbf{J}$  is given by the derivative of the reconstruction error for the  $j^{\text{th}}$  training sample with respect to the  $k^{\text{th}}$  element of the vector  $\mathbf{p}$ . We can write the resulting equations in compact form as follows

$$(\mathbf{J}^T \mathbf{J})\delta = \mathbf{J}^T \mathbf{G}(\mathbf{p}) \quad (5)$$

where  $\mathbf{G}(\mathbf{p})$  is a vector whose elements correspond to the values  $g(\mathbf{p})$  for each of the training instances, *i.e.* the diagonal coefficients of the connection map.

In [23], the increment  $\delta$  is computed devoid of the damping factor  $\beta$  by approximating the Hessian on the tangent bundle of the manifold. This yields

$$\delta = -\frac{1}{\rho} \circ \mathbf{J}^T[\mathbf{G}(\mathbf{p})] \quad (6)$$

where  $\rho$  is the product of the leading eigenpair, *i.e.* eigenvalue and eigenvector, of  $\mathbf{J}^T\mathbf{J}$  and  $\circ$  denotes the Hadamard (entry-wise) product.

### 3 Experiments

In this section, we commence by elaborating on the datasets used in our experiments. Later on, we present a quantitative analysis for our approach and illustrate its utility for band selection and spectral reconstruction.

#### 3.1 Datasets

For the experiments presented in this section, we use two widely available hyper-spectral image datasets of rural and urban environments for both, training and testing.

**NUS Dataset**<sup>1</sup>. This dataset consist of 64 images acquired using a Specim camera with a spectral resolution of 10 nm in the visible spectrum. It is worth noting that the dataset has been divided into testing and training sets. Here, all our experiments have been effected using the split as originally presented in [16]. Note that using the full set of pixels from the training images is, in practice, infeasible. As a result, for training our neural network we have randomly selected 2, 108, 000 pixel patches from the training imagery of the dataset.

**Scyllarus Series A Dataset of Spectral Images**<sup>2</sup>. This dataset consists of 73, 2 Mpx images acquired with a Liquid Crystal Tunable Filter (LCTF) tuned at intervals of 10 nm in the visible spectrum. The intensity response was recorded with a low distortion intensified 12-bit precision camera. For training and testing, we have used a tenfold random 13–60 image testing-training split. Similarly to the procedure applied to the NUS dataset, for the training involving the Scyllarus images, we have selected 230, 000 pixel patches.

#### 3.2 Settings

All the spectral reconstructions performed herein cover the range [400 nm, 700 nm] in 10 nm steps. For the computation of all the pseudocolour RGB imagery shown herein we have made use of the CIE color sensitivity functions [10]. Also, in all our experiments, we have quantified the error using both,

<sup>1</sup> The dataset can be downloaded from: [http://www.comp.nus.edu.sg/~whitebal/spectral\\_reconstruction/](http://www.comp.nus.edu.sg/~whitebal/spectral_reconstruction/).

<sup>2</sup> Downloadable at: <http://www.scyllarus.com>.



**Fig. 4.** Sample results delivered by our net trained using the Scyllarus dataset on two sample images, one from the NUS (top row) and another one from the Scyllarus dataset (bottom row). In each row, from left-to-right: Input images in pseudocolour, images delivered by our net also in pseudocolour, mean-squared difference and Euclidean angular error for the two sample images. (Color figure online)

the Euclidean angle in degrees and the absolute difference between the ground truth and the image irradiance yielded by our network. We opt for this error measure as it is widely used in previous works [21]. Note that the other error measure used elsewhere is the RMS error [16]. It is worth noting, however, that the Euclidean angle and the RMS error are correlated when the spectra is normalised to unit L2-norm. Finally, for training, all patches for both datasets are  $32 \times 32$  pixels.

### 3.3 Band Reduction Results

We commence by evaluating the capacity of our network to remove spectral bands from further consideration while being able to recover the full spectral radiance at output. To illustrate this, in Fig. 3, we show a sample spectral image from the NUS testing set whose spectra has been recovered by our network. At training, our net reduced the number of input bands from 31 to 16, *i.e.* by approximately 50%. In the figure, we show the spectra delivered by our network at testing, where the trace accounts for the mean spectral irradiance whereas the error-bars represent the variance of the spectral difference. Note that, from the plots, we can see that the spectral difference is quite small.

We provide further qualitative results on Fig. 4. In the figure, we show a sample testing image, in pseudocolour, for both datasets, *i.e.* NUS and Scyllarus, the mean-squared error and the Euclidean angle difference for the image recovered by our network using the connection map yielded by setting the upper bound of the regularisation term weight  $\lambda$  to 0.03. For the NUS image, the mean squared error is in average  $1.1 \times 10^{-3}$  with a variance of  $5.11 \times 10^{-4}$ . Similarly, the mean Euclidean angle difference in degrees is 8.34 with a variance of 3.456. For the sample Scyllarus image, the average mean-squared error and Euclidean angular

**Table 1.** Qualitative results yielded by the network using both sets for training and testing. In the table we show the mean and variance per-pixel Euclidean angle difference (in degrees) and normalised absolute band difference between the reconstruction yielded by our network and the testing ground truth imagery for different values of  $\lambda$ . The absolute lowest error per dataset is in bold font for each dataset and training set option.

Training set	Parameters		Euclidean angle (degrees)		Absolute difference	
	$\lambda$	$ T $	Scyllarus	NUS	Scyllarus	NUS
NUS	0.03	19	<b>6.17 ± 13.45</b>	<b>5.34 ± 12.53</b>	$0.0428 \pm 1.49 \times 10^{-3}$	<b>0.0159 ± 2.38 × 10<sup>-3</sup></b>
	0.05	17	7.47 ± 15.53	6.62 ± 12.97	$0.0430 \pm 1.50 \times 10^{-3}$	0.0165 ± 2.41 × 10 <sup>-3</sup>
	0.07	16	8.06 ± 16.15	7.53 ± 13.25	$0.0433 \pm 1.52 \times 10^{-3}$	0.0169 ± 2.42 × 10 <sup>-3</sup>
	0.09	14	9.98 ± 18.23	8.75 ± 14.08	$0.0461 \pm 1.54 \times 10^{-3}$	0.0173 ± 2.45 × 10 <sup>-3</sup>
Scyllarus	0.03	16	7.06 ± 15.36	8.64 ± 15.12	<b>0.0312 ± 1.50 × 10<sup>-3</sup></b>	0.0163 ± 2.55 × 10 <sup>-3</sup>
	0.05	16	7.28 ± 15.92	8.77 ± 15.26	$0.0338 \pm 1.51 \times 10^{-3}$	0.0166 ± 2.57 × 10 <sup>-3</sup>
	0.07	15	9.11 ± 15.87	9.78 ± 16.18	$0.0346 \pm 1.51 \times 10^{-3}$	0.0168 ± 2.58 × 10 <sup>-3</sup>
	0.09	14	9.23 ± 15.39	9.67 ± 16.67	$0.0382 \pm 1.54 \times 10^{-3}$	0.0172 ± 2.61 × 10 <sup>-3</sup>

difference is  $5.94 \times 10^{-3}$  and 10.81, respectively with corresponding variance values of  $3.3 \times 10^{-4}$  and 15.52.

In Table 1, we turn our attention to a more quantitative analysis of the results yielded by our approach. Recall that, as presented in Sect. 2.2, the parameter  $\lambda$  controls the influence of the regularisation term in Eq. 3. Thus, in the table, we show the angular error and the mean-squared spectral difference for the testing result on both datasets as a function of both, the value of  $\lambda$  and the dataset used for training. Note that, as expected, the network performs best when  $\lambda$  is the smallest and the training and testing data arise from the same image set. This is expected since a smaller  $\lambda$  preserves more bands, *i.e.* the regularisation is less “aggressive”. Nonetheless, as shown in our qualitative and quantitative results, the network is quite competitive even for larger values of  $\lambda$  and cross-dataset training-testing operations.

## 4 Conclusions

In this paper we have proposed a generic, content-free, non-linear mapping between a subset of wavelength indexed bands and the scene reflectance. Our approach is based on a convolutional neural network that learns the mapping of a pixel given its neighbourhood. The architecture incorporates a trainable input connection map to learn the subset of wavelengths that is relevant. Our approach does not depend on the contents of the scene nor on the camera used for acquiring the images. Our experimental results show that, once the network is trained, it is capable of recovering the spectral irradiance with a reduced number of wavelength indexed bands at input. This opens up the possibility of recovering the spectral irradiance of the scene with a much improved spectral resolution making use of a reduced number of wavelength indexed bands.

**Acknowledgment.** The authors would like to thank NVIDIA for providing the GPUs used to obtain the results shown in this paper through their Academic grant programme.

## References

1. Akgun, T., Altunbasak, Y., Mersereau, R.M.: Super-resolution reconstruction of hyperspectral images. *IEEE Trans. Image Process.* **14**(11), 1860–1875 (2005)
2. Alvarez, J.M., Salzmann, M.: Learning the number of neurons in deep networks. In: *NIPS* (2016)
3. Cariou, C., Chehdi, K., Moan, S.L.: Bandclust: an unsupervised band reduction method for hyperspectral remote sensing. *IEEE Geosci. Remote Sens. Lett.* **8**(3), 565–569 (2011)
4. Chang, J.Y., Lee, K.M., Lee, S.U.: Shape from shading using graph cuts. In: *Proceedings of the International Conference on Image Processing* (2003)
5. Ejaz, T., Horiuchi, T., Ohashi, G., Shimodaira, Y.: Development of a camera system for the acquisition of high-fidelity colors. *IEICE Trans. Electron.* **E-89C**(10), 1441–1447 (2006)
6. Finlayson, G.D., Drew, M.S.: The maximum ignorance assumption with positivity. In: *Proceedings of the IS&T/SID 4th Color Imaging Conference*, pp. 202–204 (1996)
7. Gu, L., Huynh, C.P., Robles-Kelly, A.: Material-specific user colour profiles from imaging spectroscopy data. In: *IEEE International Conference on Computer Vision* (2011)
8. Gu, L., Robles-Kelly, A., Zhou, J.: Efficient estimation of reflectance parameters from imaging spectroscopy. *IEEE Trans. Image Process.* **99**, 1 (2013)
9. Guo, B., Gunn, S.R., Damper, R.I., Nelson, J.D.B.: Band selection for hyperspectral image classification using mutual information. *IEEE Geosci. Remote Sens. Lett.* **3**(4), 522–526 (2006)
10. Judd, D.B.: Report of U.S. secretariat committee on colorimetry and artificial daylight, p. 11 (1951)
11. Kawakami, R., Zhao, H., Tan, R., Ikeuchi, K.: Camera spectral sensitivity and white balance estimation from sky images. *Int. J. Comput. Vis.* **105**(3), 187–204 (2013)
12. Koray, K., Sermanet, P., Boureau, Y.L., Gregor, K., Mathieu, M., LeCun, Y.: Learning convolutional feature hierarchies for visual recognition. In: *NIPS*, pp. 1090–1098 (2010)
13. Longere, P., Brainard, D.H.: Simulation of digital camera images from hyperspectral input. In: van den Branden Lambrecht, C. (ed.) *Vision Models and Applications to Image and Video Processing*, pp. 123–150. Kluwer (2001)
14. Marquardt, D.: An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.* **11**, 431–441 (1963)
15. Nguyen, R.M.H., Prasad, D.K., Brown, M.S.: Raw-to-raw: mapping between image sensor color responses. In: *Computer Vision and Pattern Recognition* (2014)
16. Nguyen, R.M.H., Prasad, D.K., Brown, M.S.: Training-based spectral reconstruction from a single RGB image. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) *ECCV 2014*. LNCS, vol. 8695, pp. 186–201. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-10584-0\\_13](https://doi.org/10.1007/978-3-319-10584-0_13)
17. Nocedal, J., Wright, S.: *Numerical Optimization*. Springer, Heidelberg (2000). <https://doi.org/10.1007/978-0-387-40065-5>



18. Robles-Kelly, A.: Single image spectral reconstruction for multimedia applications. In: ACM International Conference on Multimedia, pp. 251–260 (2015)
19. Sharma, G., Vrhel, M.J., Trussell, H.J.: Color imaging for multimedia. *Proc. IEEE* **86**(6), 1088–1108 (1998)
20. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014)
21. van de Weijer, J., Gevers, T., Gijsenij, A.: Edge-based color constancy. *IEEE Trans. Image Process.* **16**(9), 2207–2214 (2007)
22. Zare, A., Gader, P.: Hyperspectral band selection and endmember detection using sparsity promoting priors. *IEEE Geosci. Remote Sens. Lett.* **5**(2), 256–260 (2008)
23. Zhao, H., Robles-Kelly, A., Zhou, J., Lu, J., Yang, J.: Graph attribute embedding via riemannian submersion learning. *Comput. Vis. Image Underst.* **115**(7), 962–975 (2011)



# Local Patterns and Supergraph for Chemical Graph Classification with Convolutional Networks

Évariste Daller<sup>(✉)</sup> , Sébastien Bougleux , Luc Brun ,  
and Olivier Lézoray 

Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, Caen, France  
{evariste.daller,bougleux,olivier.lezoray}@unicaen.fr,  
luc.brun@ensicaen.fr

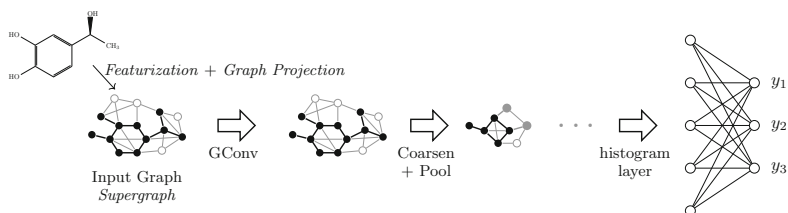
**Abstract.** Convolutional neural networks (CNN) have deeply impacted the field of machine learning. These networks, designed to process objects with a fixed topology, can readily be applied to images, videos and sounds but cannot be easily extended to structures with an arbitrary topology such as graphs. Examples of applications of machine learning to graphs include the prediction of the properties molecular graphs, or the classification of 3D meshes. Within the chemical graphs framework, we propose a method to extend networks based on a fixed topology to input graphs with an arbitrary topology. We also propose an enriched feature vector attached to each node of a chemical graph and a new layer interfacing graphs with arbitrary topologies with a full connected layer.

**Keywords:** Graph-CNNs · Graph classification · Graph edit distance

## 1 Introduction

Convolutional neural networks (CNN) [13] have deeply impacted machine learning and related fields such as computer vision. These large breakthrough encouraged many researchers [4, 5, 9, 10] to extend the CNN framework to unstructured data such as graphs, point clouds or manifolds. The main motivation for this new trend consists in extending the initial successes obtained in computer vision to other fields such as indexing of textual documents, genomics, computer chemistry or indexing of 3D models.

The initial convolution operation defined within CNN, uses explicitly the fact that objects (e.g. pixels) are embedded within a plane and on a regular grid. These hypothesis do not hold when dealing with convolution on graphs. A first approach related to the graph signal processing framework uses the link between convolution and Fourier transform as well as the strong similarities between the Fourier transform and the spectral decomposition of a graph. For example, Bruna et al. [5] define the convolution operation from the Laplacian spectrum of the graph encoding the first layer of the neural network. However this

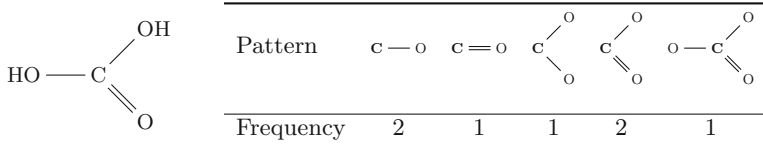


**Fig. 1.** Illustration of our propositions on a graph convolutional network

approach requires a costly decomposition into singular Laplacian values during the creation of the convolution network as well as costly matrices multiplications during the test phase. These limitations are partially solved by Defferdar et al. [9] who propose a fast implementation of the convolution based on Chebyshev polynomials (CGCNN). This implementation allows a recursive and efficient definition of the filtering operation while avoiding the explicit computation of the Laplacian. However, both methods are based on a fixed graph structure. Such networks can process different signals superimposed onto a fixed input layer but are unable to predict properties of graphs with variable topologies.

Another family of methods is based on a spatial definition of the graph convolution operation. Kipf and Welling [12] proposed a model (CGN) which approximates the local spectral filters from [9]. Using this formulation, filters are no longer based on the Laplacian but on a weight associated to each component of the vertices' features for each filter. The learning process of such weights is independent of the graph topology. Therefore graph neural networks based on this convolution scheme can predict properties of graphs with various topologies. The model proposed by Duvenaud et al. [10] for fingerprint extraction is similar to [12], but considers a set of filters for each possible degree of vertices. These last two methods both weight each components of the vertices' feature vectors. Verma et al. [17] propose to attach a weight to edges through the learning of a parametric similarity measure between the features of adjacent vertices. Similarly, Simonovsky and Komodakis [15] learn a weight associated to each edge label. Finally, Atwood and Towsley [1] (with DCNN) remove the limitation of the convolution to the direct neighborhood of each vertex by considering powers of a transition matrix defined as a normalization of the adjacency matrix by vertices' degrees. A main drawback of this non-spectral approach is that there exist intrinsically no best way to match the learned convolution weights with the elements of the receptive field, hence this variety of recent models.

In this paper, we propose to unify both spatial and spectral approaches by using as input layer a super-graph deduced from a graph train set. In addition, we propose an enriched feature vector within the framework of chemical graphs. Finally, we propose a new bottleneck layer at the end of our neural network which is able to cope with the variable size of the previous layer. These contributions are described in Sect. 2 and evaluated in Sect. 3 through several experiments.



Pattern	c — o	c = o	c $\begin{matrix} \diagup \text{o} \\ \diagdown \text{o} \end{matrix}$	c $\begin{matrix} \diagup \text{o} \\ \diagdown \text{o} \\ \text{=o} \end{matrix}$	o — c $\begin{matrix} \diagup \text{o} \\ \diagdown \text{o} \end{matrix}$
Frequency	2	1	1	2	1

Fig. 2. Frequency of patterns associated to the central node (C).

## 2 Contributions

### 2.1 From Symbolic to Feature Graphs for Convolution

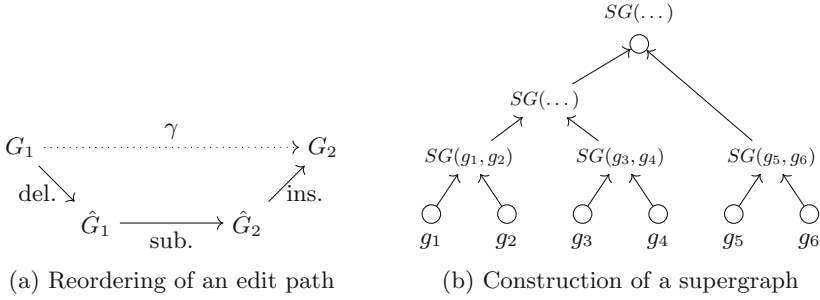
Convolution cannot be directly applied to symbolic graphs. So symbols are usually transformed into unit vectors of  $\{0, 1\}^{|\mathcal{L}|}$ , where  $\mathcal{L}$  is a set of symbols, as done in [1, 10, 15] to encode atom’s type in chemical graphs. This encoding has a main drawback, the size of convolution kernels is usually much smaller than  $|\mathcal{L}|$ . Combined with the sparsity of vectors, this produces meaningless means for dimensionality reduction. Moreover, information attached to edges is usually unused.

Let us consider a graph  $G = (V, E, \sigma, \phi)$ , where  $V$  is a set of nodes,  $E \subseteq V \times V$  a set of edges, and  $\sigma$  and  $\phi$  functions labeling respectively  $G$ ’s nodes and edges. To avoid these drawbacks, we consider for each node  $u$  of  $V$  a vector representing the distribution of small subgraphs covering this node. Let  $\mathcal{N}_u$  denotes its 1-hop neighbors. For any subset  $S \subseteq \mathcal{N}_u$ , the subgraph  $M_u^S = (\{u\} \cup S, E \cap (\{u\} \cup S) \times (\{u\} \cup S), \sigma, \phi)$  is connected (through  $u$ ) and defines a local pattern of  $u$ . The enumerations of all subsets of  $\mathcal{N}_u$  provides all local patterns of  $u$  that can be organized as a feature vector counting the number of occurrences of each local pattern. Figure 2 illustrates the computation of such a feature vector. Note that the node’s degree of chemical graphs is bounded and usually smaller than 4.

During the training phase, the patterns found for the nodes of the training graphs determine a dictionary as well as the dimension of the feature vector attached to each node. During the testing phase, we compute for each node of an input graph, the number of occurrences of its local patterns also present in the dictionary. A local pattern of the test set not present in the train set is thus discarded. In order to further enforce the compactness of our feature space, we apply a PCA on the whole set of feature vectors and project each vector onto a subspace containing 95% (fixed threshold) of the initial information.

### 2.2 Supergraph as Input Layer

As mentioned in Sect. 1, methods based on spectral analysis [5, 9] require a fixed input layer. Hence, these methods can only process functions defined on a fixed graph topology (e.g. node’s classification or regression tasks) and cannot be used to predict global properties of topologically variable graphs. We propose to remove this restriction by using as an input layer a supergraph deduced from graphs of a training set.



**Fig. 3.** Construction of a supergraph (b) using common subgraphs induced by the graph edit distance (a).

A *common supergraph* of two graphs  $G_1$  and  $G_2$  is a graph  $S$  so that both  $G_1$  and  $G_2$  are isomorphic to a subgraph of  $S$ . More generally, a common supergraph of a set of graphs  $\mathcal{G} = \{G_k = (V_k, E_k, \sigma_k, \phi_k)\}_{k=1}^n$  is a graph  $S = (V_S, E_S, \sigma_S, \phi_S)$  so that any graph of  $\mathcal{G}$  is isomorphic to a subgraph of  $S$ . So, given any two complementary subsets  $\mathcal{G}_1, \mathcal{G}_2 \subseteq \mathcal{G}$ , with  $\mathcal{G}_1 \cup \mathcal{G}_2 = \mathcal{G}$ , it holds that a supergraph of a supergraph of  $\mathcal{G}_1$  and a supergraph of  $\mathcal{G}_2$  is a supergraph of  $\mathcal{G}$ . The latter can thus be defined by applying this property recursively on the subsets. This describes a tree hierarchy of supergraphs, rooted at a supergraph of  $\mathcal{G}$ , with the graphs of  $\mathcal{G}$  as leaves. We present a method to construct hierarchically a supergraph so that it is formed of a minimum number of elements.

A common supergraph  $S$  of two graphs, or more generally of  $\mathcal{G}$ , is a *minimum common supergraph* (MCS) if there is no other supergraph  $S'$  of  $\mathcal{G}$  with  $|V_{S'}| < |V_S|$  or  $(|V_{S'}| = |V_S|) \wedge (|E_{S'}| < |E_S|)$ . Constructing such a supergraph is difficult and can be linked to the following notion. A *maximum common subgraph* (mcs) of two graphs  $G_k$  and  $G_l$  is a graph  $G_{k,l}$  that is isomorphic to a subgraph  $\hat{G}_k$  of  $G_k$  and to a subgraph  $\hat{G}_l$  of  $G_l$ , and so that there is no other common subgraph  $G'$  of both  $G_k$  and  $G_l$  with  $|V_{G'}| > |V_{G_{k,l}}|$  or  $(|V_{G'}| = |V_{G_{k,l}}|) \wedge (|E_{G'}| > |E_{G_{k,l}}|)$ . Then, given a maximum common subgraph  $G_{k,l}$ , the graph  $S$  obtained from  $G_{k,l}$  by adding the elements of  $G_k$  not in  $\hat{G}_k$  and the elements of  $G_l$  not in  $\hat{G}_l$  is a minimum common supergraph of  $G_k$  and  $G_l$ . This property shows that a minimum common supergraph can thus be constructed from a maximum common subgraph. These notions are both related to the notion of error-correcting graph matching and graph edit distance [6].

The *graph edit distance* (GED) captures the minimal amount of distortion needed to transform an attributed graph  $G_k$  into an attributed graph  $G_l$  by iteratively editing both the structure and the attributes of  $G_k$ , until  $G_l$  is obtained. The resulting sequence of *edit operations*  $\gamma$ , called *edit path*, transforms  $G_k$  into  $G_l$ . Its cost (the strength of the global distortion) is measured by  $L_c(\gamma) = \sum_{o \in \gamma} c(o)$ , where  $c(o)$  is the cost of the edit operation  $o$ . Among all edit paths from  $G_k$  to  $G_l$ , denoted by the set  $\Gamma(G_k, G_l)$ , a *minimal-cost edit path* is a path having a minimal cost. The GED from  $G_k$  to  $G_l$  is defined as the cost of a minimal-cost edit path:  $d(G_k, G_l) = \min_{\gamma \in \Gamma(G_k, G_l)} L_c(\gamma)$ .

Under mild constraints on the costs [3], an edit path can be organized into a succession of removals, followed by a sequence of substitutions and ended by a sequence of insertions. This reordered sequence allows to consider the subgraphs  $\hat{G}_k$  of  $G_k$  and  $\hat{G}_l$  of  $G_l$ . The subgraph  $\hat{G}_k$  is deduced from  $G_k$  by a sequence of node and edge removals, and the subgraph  $\hat{G}_l$  is deduced from  $\hat{G}_k$  by a sequence of substitutions (Fig. 3a). By construction,  $\hat{G}_k$  and  $\hat{G}_l$  are structurally isomorphic, and an *error-correcting graph matching* (ECGM) between  $G_k$  and  $G_l$  is a bijective function  $f : \hat{V}_k \rightarrow \hat{V}_l$  matching the nodes of  $\hat{G}_k$  onto the ones of  $\hat{G}_l$  (correspondences between edges are induced by  $f$ ).

Then ECGM, mcs and MCS are related as follows. For specific edit cost values [6] (not detailed here), if  $f$  corresponds to an optimal edit sequence, then  $\hat{G}_k$  and  $\hat{G}_l$  are mcs of  $G_k$  and  $G_l$ . Moreover, adding to a mcs of  $G_k$  and  $G_l$  the missing elements from  $G_k$  and  $G_l$  leads to an MCS of these two graphs. We use this property to build the global supergraph of a set of graphs.

**Supergraph Construction.** The proposed hierarchical construction of a common supergraph of a set of graphs  $\mathcal{G} = \{G_i\}_i$  is illustrated by Fig. 3b. Each level  $k$  of the hierarchy contains  $N_k$  graphs. They are merged by pairs to produce  $\lfloor N_k/2 \rfloor$  supergraphs. In order to restrain the size of the final supergraph, a natural heuristic consists in merging close graphs according to the graph edit distance. This can be formalized as the computation of a maximum matching  $M^*$ , in the complete graph over the graphs of  $\mathcal{G}$ , minimizing:

$$M^* = \arg \min_M \sum_{(g_i, g_j) \in M} d(g_i, g_j) \quad (1)$$

where  $d(\cdot, \cdot)$  denotes the graph edit distance. An advantage of this kind of construction is that it is highly parallelizable. Nevertheless, computing the graph edit distance is NP-hard. Algorithms that solve the exact problem cannot be reasonably used here. So we considered a bipartite approximation of the GED [14] to compute  $d(\cdot, \cdot)$  and solve (1), while supergraphs are computed using a more precise but more computationally expansive algorithm [7].

### 2.3 Projections as Input Data

The supergraph computed in the previous section can be used as an input layer of a graph convolutional neural network based on spectral graph theory [5, 9] (Sect. 1). Indeed, the fixed input layer allows to consider convolution operations based on the Laplacian of the input layer. However, each input graph for which a property has to be predicted, must be transformed into a signal on the supergraph. This last operation is allowed by the notion of projection, a side notion of the graph edit distance.

**Definition 1 (Projection).** Let  $f$  be an ECGM between two graphs  $G$  and  $S$  and let  $(\hat{V}_S, \hat{E}_S)$  be the subgraph of  $S$  defined by  $f$  (Fig. 3). A projection of  $G = (V, E, \sigma, \phi)$  onto  $S = (V_S, E_S, \sigma_S, \phi_S)$  is a graph  $P_S^f(G) = (V_S, E_S, \sigma_P, \phi_P)$  where  $\sigma_P(u) = (\sigma \circ f^{-1})(u)$  for any  $u \in \hat{V}_S$  and 0 otherwise. Similarly,  $\phi_P(\{u, v\}) = \phi(\{f^{-1}(u), f^{-1}(v)\})$  for any  $\{u, v\}$  in  $\hat{E}_S$  and 0 otherwise.

Let  $\{G_1, \dots, G_n\}$  be a graph training set and  $S$  its the associated supergraph. The projection  $P_S^f(G_i)$  of a graph  $G_i$  induces a signal on  $S$  associated to a value to be predicted. For each node of  $S$  belonging to the projection of  $G_i$ , this signal is equal to the feature vector of this node in  $G_i$ . This signal is null outside the projection of  $G_i$ . Moreover, if the edit distance between  $G_i$  and  $S$  can be computed through several edit paths with a same cost (i.e., several ECGM  $f_1, \dots, f_m$ ), the graph  $G_i$  will be associated to these projections  $P_S^{f_1}(G_i), \dots, P_S^{f_m}(G_i)$ . Remark that a graph belonging to a test dataset may also have several projections. In this case, it is mapped onto the majority class among its projections. A natural data augmentation can thus be obtained by learning  $m$  equivalent representations of a same graph on the supergraph, associated to the same value to be predicted. Note that this data augmentation can also be increased by considering  $\mu m$  non-minimal ECGM, where  $\mu$  is a parameter. To this end, we use [7] to compute a set of non-minimal ECGM between an input graph  $G_i$  and the supergraph  $S$  and we sort this set increasingly according to the cost of the associated edit paths.

## 2.4 Bottleneck Layer with Variable Input Size

A multilayer perceptron (MLP), commonly used in the last part of multilayer networks, requires that the previous layer has a fixed size and topology. Without the notion of supergraph, this last condition is usually not satisfied. Indeed, the size and topology of intermediate layers are determined by those of the input graphs, which generally vary. Most of graph neural networks avoid this drawback by performing a global pooling step through a bottleneck layer. This usually consists in averaging the components of the feature vectors across the nodes of the current graph, the so-called global average pooling (GAP). If for each node  $v \in V$  of the previous layer, the feature vector  $h(v) \in \mathbb{R}^D$  has a dimension  $D$ , GAP produces a mean vector  $(\frac{1}{|V|} \sum_{v \in V} h_c(v))_{c=1, \dots, D}$  describing the graph globally in the feature space.

We propose to improve the pooling step by considering the distribution of feature activations across the graph. A simple histogram can not be used here, due to its non-differentiability, differentiability being necessary for backpropagation. To guarantee this property holds, we propose to interpolate the histogram by using averages of Gaussian activations. For each component  $c$  of a given a feature vector  $h(v)$ , the height of a bin  $k$  of this pseudo-histogram is computed as follows:

$$b_{ck}(h) = \frac{1}{|V|} \sum_{v \in V} \exp\left(\frac{-(h_c(v) - \mu_{ck})^2}{\sigma_{ck}^2}\right) \quad (2)$$

The size of the layer is equal to  $D \times K$ , where  $K$  is the number of bins defined for each component.

In this work, the parameters  $\mu_{ck}$  and  $\sigma_{ck}$  are fixed and not learned by the network. To choose them properly, the model is trained with a GAP layer for few iterations (10 in our experiments), then it is replaced by the proposed layer. The weights of the network are preserved, and the parameters  $\mu_{ck}$  are uniformly spread between the minimum and the maximum values of  $h_c(v)$ . The parameters

$\sigma_{ck}$  are fixed to  $\sigma_{ck} = \delta_\mu/3$  with  $\delta_\mu = \mu_{ci+1} - \mu_{ci}$ ,  $\forall 1 \leq i < K$ , to ensure an overlap of the Gaussian activations.

Since this layer has no learnable parameters, the weights  $\alpha_c(i)$  of the previous layer  $h$  are adjusted during the backpropagation for every node  $i \in V$ , according to the partial derivatives of the loss function  $L$ :  $\frac{\partial L}{\partial \alpha_c(i)} = \frac{\partial L}{\partial b_{ck}(h)} \frac{\partial b_{ck}(h)}{\partial h_c(i)} \frac{\partial h_c(i)}{\partial \alpha_c(i)}$ . The derivative of the bottleneck layer w.r.t. its input is given by:

$$\forall i \in V, \quad \frac{\partial b_{ck}(h)}{\partial h_c(i)} = \frac{-2(h_c(i) - \mu_{ck})}{|V|\sigma_{ck}^2} \exp\left(\frac{-(h_c(i) - \mu_{ck})^2}{\sigma_{ck}^2}\right). \quad (3)$$

It lies between  $-\frac{\sqrt{2}}{|V|\sigma_{ck}}e^{-1/2}$  and  $\frac{\sqrt{2}}{|V|\sigma_{ck}}e^{-1/2}$ .

### 3 Experiments

We compared the behavior of several graph convolutional networks, with and without the layers presented in the previous section, for the classification of chemical data encoded by graphs. The following datasets were used: NCI1, MUTAG, ENZYMES, PTC, and PAH. Table 1 summarizes their main characteristics. NCI1 [18] contains 4110 chemical compounds, labeled according to their capacity to inhibit the growth of certain cancerous cells. MUTAG [8] contains 188 aromatic and heteroaromatic nitrocompounds, the mutagenicity of which has to be predicted. ENZYMES [2] contains 600 proteins divided into 6 classes of enzymes (100 per class). PTC [16] contains 344 compounds labeled as carcinogenic or not for rats and mice. PAH<sup>1</sup> contains non-labeled cyclic carcinogenic and non-carcinogenic molecules.

#### 3.1 Baseline for Classification

We considered three kinds of graph convolutional networks. They differ by the definition of their convolutional layer. CGCNN [9] is a deep network based on a pyramid of reduced graphs. Each reduced graph corresponds to a layer of the network. The convolution is realized by spectral analysis and requires the computation of the Laplacian of each reduced graph. The last reduced graph is followed by a fully connected layer. GCN [12] and DCNN [1] networks do not use spectral analysis and are referred to as spatial networks. GCN can be seen as an approximation of [9]. Each convolutional layer is based on  $F$  filtering operations associating a weight to each component of the feature vectors attached to nodes. These weighted vectors are then combined through a local averaging. DCNN [1] is a nonlocal model in which a weight on each feature is associated to a hop  $h < H$  and hence to a distance to a central node ( $H$  is thus the radius of a ball centered on this central node). The averaging of the weighted feature vectors is then performed on several hops for each node.

To measure the effects of our contributions when added to the two spatial networks (DCNN and GCN), we considered several versions obtained as follows

<sup>1</sup> PAH is available at: <https://iapr-tc15.greyc.fr/links.html>.



**Table 1.** Characteristics of datasets.  $V$  and  $E$  denotes resp. nodes and edges sets of the datasets’ graphs, while  $V_S$  and  $E_S$  denotes nodes and edges sets of the datasets’ supergraphs

	NCI1	MUTAG	ENZYMES	PTC	PAH
#graphs	4110	188	600	344	94
mean $ V $ , mean $ E $	(29.9, 32.3)	(17.9, 19.8)	(32.6, 62.1)	(14.3, 14.7)	(20.7, 24.4)
mean $ V_S $	192.8	42.6	177.1	102.6	26.8
mean $ E_S $	4665	146	1404	377	79
#labels, #patterns	(37, 424)	(7, 84)	(3, 240)	(19, 269)	(1, 4)
#classes	2	2	6	2	2
#positive, #negative	(2057, 2053)	(125, 63)	–	(152, 192)	(59, 35)

(Table 2). We used two types of characteristics attached to the nodes of the graphs (input layer): characteristics based on the canonical vectors of  $\{0, 1\}^{|\mathcal{L}|}$  as in [1, 10, 15], and those based on the patterns proposed in Sect. 1. Note that PAH has few different patterns (Table 1), PCA was therefore not applied to this data to reduce the size of features. Since spatial networks can handle arbitrary topology graphs, the use of a supergraph is not necessary. However, since some nodes have a null feature in a supergraph (Definition 1), a convolution performed on a graph gives results different from those obtained by a similar convolution performed on the projection of the graph on a supergraph. We hence decided to test spatial networks with a supergraph. For the other network (CGCNN), we used the features based on patterns and a supergraph.

For the architecture of spatial networks, we followed the one proposed by [1], with a single convolutional layer. For CGCNN we used two convolutional layers to take advantage of the coarsening as it is part of this method. For DCNN,  $H = 4$ . For CGCNN and GCN,  $F = 32$  filters were used. The optimization was achieved by Adam [11], with at most 500 epochs and early stopping. The experiments were done in 10 fold cross-validation which required to compute the supergraphs of all training graphs. Datasets were augmented by 20% of non-minimal cost projections with the method described in Sect. 2.3.

### 3.2 Discussion

As illustrated in Table 2, the features proposed in Sect. 2.1 improve the classification rate in most cases. For some datasets, the gain is higher than 10% points. The behavior of the two spatial models (DCNN and GCN) is also improved, for every dataset, by replacing global average pooling by the histogram bottleneck layer described in Sect. 2.4. These observations point out the importance of the global pooling step for these kind of networks.

Using a supergraph as an input layer (column s-g) opens the field of action of spectral graph convolutional networks to graphs with different topologies, which is an interesting result in itself. Results are comparable to the ones obtained with the other methods (improve the baseline models with no histogram layer), but

**Table 2.** Mean accuracy (10-fold cross validation) of graph classification by three networks (GConv), with the features proposed in Sect. 2.1 (*feat.*) and the supergraph (*s-g*). Global pooling (gpool) is done using global average pooling (GAP) or with histogram bottleneck layer (hist).

GConv	feat.	s-g	gpool	NCI1	MUTAG	ENZYMES	PTC	PAH
DCNN	–	–	GAP	62.61	66.98	18.10	56.60	57.18
	✓	–	GAP	67.81	81.74	31.25	59.04	54.70
	✓	–	hist	71.47	82.22	38.55	<b>60.43</b>	66.90
	✓	✓	hist	<b>73.95</b>	<b>83.57</b>	<b>40.83</b>	56.04	<b>71.35</b>
GCN	–	–	GAP	55.44	70.79	16.60	52.17	63.12
	✓	–	GAP	66.39	82.22	32.36	58.43	57.80
	✓	–	hist	<b>74.76</b>	<b>82.86</b>	37.90	<b>62.78</b>	<b>72.80</b>
	✓	✓	hist	73.02	80.44	<b>46.23</b>	61.60	71.50
CGCNN	✓	✓	–	68.36	75.87	33.27	60.78	63.73

this is a first result for these networks for the classification of graphs. The sizes of supergraphs reported in Table 1 remain reasonable regarding the number of graphs and the maximum size in each dataset. Nevertheless, this strategy only enlarge each data up to the supergraph size.

## 4 Conclusions

We proposed features based on patterns to improve the performances of graph neural networks on chemical graphs. We also proposed to use a supergraph as input layer in order to extend graph neural networks based on spectral theory to the prediction of graph properties for arbitrary topology graphs. The supergraph can be combined with any graph neural network, and for some datasets the performances of graph neural networks not based on spectral theory were improved. Finally, we proposed an alternative to the global average pooling commonly used as bottleneck layer in the final part of these networks.

## References

1. Atwood, J., Towsley, D.: Diffusion-convolutional neural networks. Adv. Neural Inf. Process. Syst. **29**, 2001–2009 (2016)
2. Borgwardt, K.M., Ong, C.S., Schönauer, S., Vishwanathan, S.V.N., Smola, A.J., Kriegel, H.P.: Protein function prediction via graph kernels. Bioinformatics **21**(suppl 1), i47–i56 (2005). <https://doi.org/10.1093/bioinformatics/bti1007>
3. Bogleux, S., Brun, L., Carletti, V., Foggia, P., Gaüzère, B., Vento, M.: Graph edit distance as a quadratic assignment problem. Pattern Recogn. Lett. **87**, 38–46 (2017). <https://doi.org/10.1016/j.patrec.2016.10.001>

4. Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: going beyond euclidean data. *IEEE Sig. Process. Mag.* **34**(4), 18–42 (2017). <https://doi.org/10.1109/MSP.2017.2693418>
5. Bruna, J., Zaremba, W., Szlam, A., Lecun, Y.: Spectral networks and deep locally connected networks on graphs. Technical report (2014). [arXiv:1312.6203v2](https://arxiv.org/abs/1312.6203v2) [cs.LG]
6. Bunke, H., Jiang, X., Kandel, A.: On the minimum common supergraph of two graphs. *Computing* **65**(1), 13–25 (2000). <https://doi.org/10.1007/PL00021410>
7. Daller, É., Bougleux, S., Gaüzère, B., Brun, L.: Approximate graph edit distance by several local searches in parallel. In: *Proceedings of ICPRAM 2018* (2018). <https://doi.org/10.5220/0006599901490158>
8. Debnath, A., Lopez de Compadre, R.L., Debnath, G., Shusterman, A., Hansch, C.: Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. *J. Med. Chem.* **34**, 786–797 (1991). <https://doi.org/10.1021/jm00106a046>
9. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. *Adv. Neural Inf. Process. Syst.* **29**, 3844–3852 (2016)
10. Duvenaud, D., et al.: Convolutional networks on graphs for learning molecular fingerprints. *Adv. Neural Inf. Process. Syst.* **28**, 2224–2232 (2015)
11. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. *CoRR* abs/1412.6980 (2014)
12. Kipf, T., Welling, M.: Semi-supervised classification with graph convolutional networks. In: *International Conference on Learning Representations* (2017)
13. LeCun, Y., Bengio, Y.: The handbook of brain theory and neural networks. Chapter Convolutional Networks for Images, Speech, and Time Series, pp. 255–258 (1998)
14. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**, 950–959 (2009). <https://doi.org/10.1016/j.imavis.2008.04.004>
15. Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2017). <https://doi.org/10.1109/cvpr.2017.11>
16. Toivonen, H., Srinivasan, A., King, R., Kramer, S., Helma, C.: Statistical evaluation of the predictive toxicology challenge 2000–2001. *Bioinformatics* **19**, 1179–1182 (2003). <https://doi.org/10.1093/bioinformatics/btg130>
17. Verma, N., Boyer, E., Verbeek, J.: FeaStNet: feature-steered graph convolutions for 3D shape analysis. In: *IEEE Conference on Computer Vision and Pattern Recognition* (2018)
18. Wale, N., Watson, I.A., Karypis, G.: Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowl. Inf. Syst.* **14**(3), 347–375 (2008). <https://doi.org/10.1109/icdm.2006.39>



# Learning Deep Embeddings via Margin-Based Discriminate Loss

Peng Sun<sup>(✉)</sup>, Wenzhong Tang, and Xiao Bai

School of Computer Science and Engineering and Beijing Advanced Innovation,  
Center for Big Data and Brain Computing, Beihang University, Beijing, China  
{pengsun, tangwenzhong, baixiao}@buaa.edu.cn

**Abstract.** Deep metric learning has gained much popularity in recent years, following the success of deep learning. However, existing frameworks of deep metric learning based on contrastive loss and triplet loss often suffer from slow convergence, partially because they employ only one positive example and one negative example while not interacting with the other positive or negative examples in each update. In this paper, we firstly propose the strict discrimination concept to seek an optimal embedding space. Based on this concept, we then propose a new metric learning objective called Margin-based Discriminate Loss which tries to keep the similar and the dissimilar strictly discriminate by pulling multiple positive examples together while pushing multiple negative examples away at each update. Importantly, it doesn't need expensive sampling strategies. We demonstrate the validity of our proposed loss compared with the triplet loss as well as other competing loss functions for a variety of tasks on fine-grained image clustering and retrieval.

**Keywords:** Metric learning · Deep embedding  
Representation learning · Neural networks

## 1 Introduction

Metric learning for computer vision aims at finding appropriate similarity measurements between pairs of images that preserve distance structure. A good similarity can improve the performance of image search, particularly when the number of categories is very large [12] or unknown. The goal of classical metric learning methods is to find a better Mahalanobis distance in linear space. However, linear transformation has a limited number of parameters and cannot model high-order correlations between the original data dimensions. With the ability of directly learning non-linear feature representations, deep metric learning has achieved promising results on various tasks, such as face recognition [16, 17], feature matching [9, 18], visual product search [13–15], fine-grained image classification [19, 20], collaborative filtering [11, 22] and zero-shot learning [10, 21].

A wide variety of formulations have been proposed. Traditionally, these formulations encode a notion of similar and dissimilar data points. For example, contrastive loss [23], which is defined for a pair of either similar or dissimilar data points. Another commonly used family of losses is triplet loss [5], which is defined by a triplet of data points: an anchor point, and a similar and dissimilar data points. The goal in a triplet loss is to learn a distance in which the anchor point is closer to the similar point than to the dissimilar one. Although yielding promising progress, such frameworks often suffer from slow convergence and poor local optima and their effects heavily depend on sampling strategies. Hard negative data mining [5] could alleviate the problem, but it is expensive to evaluate embedding vectors in deep learning framework during hard negative example search.

To circumvent these issues, we firstly propose the strict discrimination concept to seek the optimal embedding space on the entire database. Based on this concept, we then propose a new metric learning objective called Margin-based Discriminate Loss which aims to keep similar examples and dissimilar examples strictly discriminate. The proposed loss function pulls more than one positive examples together while pushing more than one negative examples away at a time. Our method doesn't require the training data to be preprocessed in any rigid format. The proposed method is extensively evaluated on three benchmark datasets and the results show its superiority to several other state-of-the-art methods.

## 2 Related Works

### 2.1 Triplet Loss

The goal of triplet loss [5] is to push away the negative point  $x^-$  from the anchor  $x$  by a distance margin  $m_0 > 0$  compared to the positive  $x^+$ .

$$L_{triplet}(\{x, x^+, x^-\}; f(\cdot; \Theta)) = \max\{0, m_0 + \|f - f^+\|_2^2 - \|f - f^-\|_2^2\} \quad (1)$$

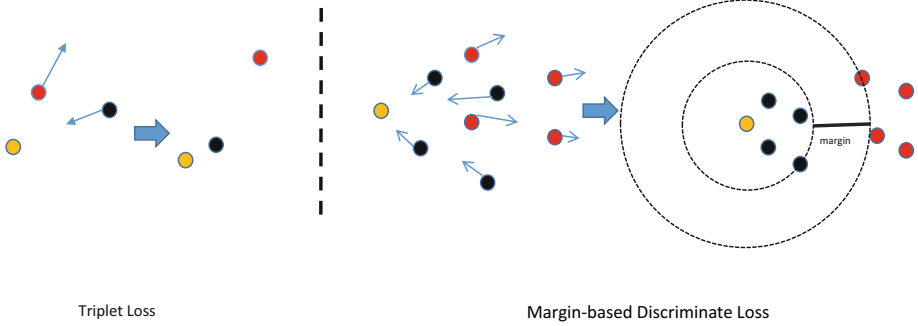
where  $f, f^+, f^-$  denote the deep embedding vector of  $x, x^+, x^-$  respectively.

### 2.2 Lifted Structured Embedding

Song et al. [3] proposed lifted structured embedding where each positive pair compares the distances against all the negative pairs weighted by the margin constraint violation. The idea is to have a differentiable smooth loss which incorporates the online hard negative mining functionality using the log-sum-exp formulation.

$$L = \frac{1}{2|P|} \sum_{(i,j) \in P} \max(0, j_{i,j})^2 \quad (2)$$

$$j_{i,j} = \log\left(\sum_{(i,k) \in N} \exp\{m_0 - D_{i,k}\} + \sum_{(j,l) \in N} \exp\{m_0 - D_{j,l}\}\right) + D_{i,j}$$



**Fig. 1.** Deep metric learning with triplet loss (left) and margin-based discriminate loss (right). The yellow, the black and the red stands for the anchor, the positive and the negative respectively. Triplet loss pulls positive example while pushing one negative example at a time. However, margin-based discriminate loss tries to keep a strict margin between the positive and the negative so as to get the optimal distribution with a minimum constraint by pulling multiple positive examples while jointly pushing multiple negative examples. (Color figure online)

where  $P$  denotes the set of pairs of examples with the same class label,  $N$  indicates the set of pairs of examples with different labels and  $D$  denotes Euclidean distance between examples.

### 2.3 N-Pair Loss

Sohn et al. [4] extended the triplet loss into N-pair loss, which significantly improves upon the triplet loss by pushing away multiple negative examples jointly at each update.

$$L_{N\text{-pair}}(\{x, x^+, \{x_i\}_{i=1}^{N-1}\}; f(\cdot; \Theta)) = \log(1 + \sum_{i=1}^{N-1} \exp(f^\top f_i - f^\top f^+)) \quad (3)$$

## 3 Margin-Based Discriminate Loss

Inspired by the max-min margin for the optimal classification plane in Support Vector Machines (SVM) [2], we want to utilize margin constraint to seek an optimal embedding space to preserve similarity structure. In the optimal embedding space, the distribution of the embedding vectors should at least have the following property. For each data point, similar points and dissimilar points should be strictly separated, which prevents that the dissimilar points are mistaken for the similar ones. Importantly, it means that no errors happen in the following tasks such as retrieval, clustering, etc. Precisely, it means that, as depicted in Fig. 1, the distance between the closest negative data point and the anchor is at least

$m_0$  greater than the distance between the farthest positive data point and the anchor.

$$\min\{d(f, f_j^-)\}_{j=1}^{n_j} - \max\{d(f, f_i^+)\}_{i=1}^{n_i} \geq m_0 \quad (4)$$

where  $d(x, y) = \|x - y\|_2^2$ , the positive constant  $m_0$  denotes the margin distance, and  $n_i$  and  $n_j$  are the number of the positive  $x^+$  and the negative  $x^-$  respectively. To enforce the above constraint, a common relaxation of Eq. 4 is the minimization of the following hinge loss,

$$\begin{aligned} L(x, \{x_i^+\}_{i=1}^{n_i}, \{x_j^-\}_{j=1}^{n_j}; f(\cdot; \Theta)) \\ = \max\{0, m_0 + \max\{d(f, f_i^+)\}_{i=1}^{n_i} - \min\{d(f, f_j^-)\}_{j=1}^{n_j}\} \end{aligned} \quad (5)$$

where  $\Theta$  are deep network parameters.

If we directly mine the hardest negative(positive) with nested min(max) functions during the training phase, the network parameters are updated only based on the similarity relations between three examples (the anchor, the hardest positive and the hardest negative). In that case, the other examples may not jointly change to make the loss (Eq. 5) decrease after each update, which is greatly unstable to learn the optimal embedding. And, empirically, it is a poor choice because the network usually converges to a bad local optimum in practice. To circumvent the issue, we replace max/min function with their smooth upper bounds which can make the loss (Eq. 5) decrease steadily by imposing constraints on multiple examples.

$$\begin{aligned} \frac{1}{K} \ln \sum_{i=1}^n \exp(Kx_i) - \max\{x_i\}_{i=1}^n \\ = \frac{1}{K} \ln(1 + \sum_{i \neq i_{max}} \exp(K(x_i - \max\{x_i\}_{i=1}^n))) \\ \leq \frac{1}{K} \ln n \end{aligned} \quad (6)$$

where the parameter  $K$  controls the approximate degree. Eq. 6 is always greater than 0 and  $\frac{1}{K} \ln \sum_{i=1}^n \exp(Kx_i)$  is a compact upper bound of  $\max\{x_i\}_{i=1}^n$ .

$$\max\{x_i\}_{i=1}^n < \frac{1}{K} \ln \sum_{i=1}^n \exp(Kx_i) \quad (7)$$

According to Eq. 7, we can derive the following.

$$-\min\{x_i\}_{i=1}^n = \max\{-x_i\}_{i=1}^n < \frac{1}{K} \ln \sum_{i=1}^n \exp(-Kx_i) \quad (8)$$

Hence we can derive the smooth upper bound of the loss function by substituting the max and min functions in Eq. 5 as follows.

$$\begin{aligned} L < \ln(1 + \exp\{m_0 + \max\{d(f, f_i^+)\}_{i=1}^{n_i} - \min\{d(f, f_j^-)\}_{j=1}^{n_j}\}) \\ < \ln(1 + \frac{e^{m_0}}{K^2} \sum_{i=1}^{n_i} \exp(K\|f - f_i^+\|_2^2) \sum_{j=1}^{n_j} \exp(-K\|f - f_j^-\|_2^2)) \end{aligned} \quad (9)$$

In this way, the loss function pulls  $n_i$  positive examples together while pushing  $n_j$  negative examples away at a time. Compared with triplet loss, it preserves the similarity structure of much more than three examples. Intuitively, the more examples are taken into account, the more global structure the loss function is aware of. Then the upper bound is used as loss function to optimize. To make full use of the batch, we rewrite the loss function to enhance the mini-batch optimization.

$$L = \sum_{m=1}^M \ln\left(1 + \frac{e^{m_0}}{K^2} \sum_{i=1}^{n_{m_i}} \exp(K\|f_m - f_i^+\|_2^2) \sum_{j=1}^{n_{m_j}} \exp(-K\|f_m - f_j^-\|_2^2)\right) \quad (10)$$

where  $M$  is the batch size. It seems that the computation is complicated. To alleviate the problem, we construct the dense pairwise squared distance matrix  $D^2$  efficiently by computing,  $D^2 = \tilde{x}1^\top + 1\tilde{x}^\top - 2XX^\top$ , where  $X \in R^{m \times d}$  denotes a batch of  $d$ -dimensional embedded features and  $\tilde{x} = [\|f(x_1)\|_2^2, \dots, \|f(x_m)\|_2^2]^\top$  indicates the column vector of squared norm of individual batch elements.

**Relation to Npair loss [4]:** Surprisingly, we find that N-pair Loss is the special case of the proposed loss. When inner product is selected as the similarity measure rather than Euclidean distance, Eq. 5 can be rewritten as  $L = \max\{0, m_0 + \max\{f^\top f_j^-\}_{j=1}^{n_j} - \min\{f^\top f_i^+\}_{i=1}^{n_i}\}$ . Following the previous analysis, the margin-based discriminate loss can be derived as follows.

$$L = \ln\left(1 + \frac{e^{m_0}}{K^2} \sum_{i=1}^{n_i} \exp(-Kf^\top f_i^+) \sum_{j=1}^{n_j} \exp(Kf^\top f_j^-)\right) \quad (11)$$

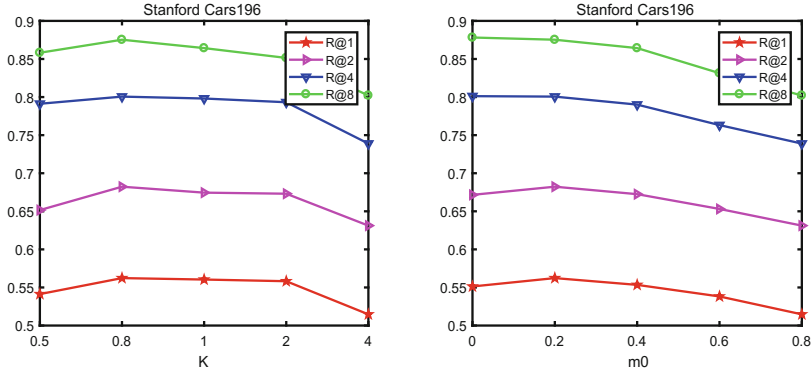
When  $m_0 = 0$ ,  $K = 1$  and  $n_i = 1$ , Npair loss function (Eq. 3) can be derived from Eq. 11.

## 4 Implementation Details

We used the Tensorflow [23] package for all methods. For the embedding vector, we  $\ell_2$  normalize the embedding vectors before computing the loss for our method. The model slightly underperformed when the embedding normalization is omitted. For fair comparison. We use the ResNet-50 architecture with batch normalization [24] pretrained on ILSVRC 2012-CLS data [25] and finetuned the network on the tested datasets. The inputs are first resized to  $256 \times 256$  pixels, and then randomly cropped to  $227 \times 227$ . For the data augmentation, we used random crop with random horizontal mirroring for training and a single center crop for testing. The experimental ablation study reported in [3] suggested that the embedding size doesn't play a crucial role during training and testing phase so we decide to set the size of the learned embeddings to 64 throughout the experiment. We use the RMSprop optimizer with the margin multiplier constant  $\gamma$  decayed at a rate of 0.94.

The proposed method does not require the data to be prepared in any rigid paired format (pairs, triplets, n-pair tuples, etc.). The proposed method just





**Fig. 2.** Comparison of different values for  $K$  and  $m_0$  for our method on Stanford cars196 dataset [8].

**Table 1.** Clustering and recall performance on CUB-200-2011 [7].

Method	Clustering	Recall@R			
	NMI	R = 1	R = 2	R = 4	R = 8
Triplet semihard	56.39	43.35	55.69	66.58	77.69
Lifted struct	57.53	44.56	56.86	68.23	79.58
Npairs	58.20	46.23	58.63	69.53	79.52
Ours	<b>59.18</b>	<b>48.53</b>	<b>59.59</b>	<b>71.24</b>	<b>81.87</b>

requires each example to have at least one positive example and one negative example in a batch. So we randomly sample  $P = 64$  groups of examples. Each group is comprised of  $Q = 4$  examples with the same class label and different groups have different class labels. Obviously, the batch size is  $M = P \times Q = 256$ . For fair comparison, we use the same batch size in the other methods.

## 5 Experiments

We evaluate deep metric learning algorithms on both image retrieval and clustering tasks on three datasets: CUB200-2011 [7], Stanford Online Products [3], and Stanford Cars196 [8]. CUB-200-2011 [7] dataset has 200 species of birds with 11,788 images included, where the first 100 species (5,864 images) are used for training and the remaining 100 species (5,924 images) are used for testing. Online Products [3] dataset contains 22,634 classes with 120,053 product images in total, where the first 11,318 classes (59,551 images) are used for training and the rest classes (60,502 images) are used for testing. Stanford Car [8] dataset is composed by 16,185 cars images of 196 classes. We use the first 98 classes (8,054 images) for training and the other 98 classes (8,131 images) for testing. Clustering quality is evaluated using the Normalized Mutual Information measure

**Table 2.** Clustering and recall performance on Stanford Online Products [3].

Method	Clustering	Recall@R		
	NMI	R = 1	R = 10	R = 100
Triplet semihard	89.35	66.65	81.36	90.56
Lifted struct	88.65	62.39	80.36	91.36
Npairs	89.16	66.42	82.69	92.69
Ours	<b>89.43</b>	<b>66.83</b>	<b>83.12</b>	<b>93.21</b>

**Table 3.** Clustering and recall performance on Stanford Cars196 [8].

Method	Clustering	Recall@R			
	NMI	R = 1	R = 2	R = 4	R = 8
Triplet semihard	53.36	51.54	63.56	73.45	82.43
Lifted struct	56.86	52.86	65.53	76.12	84.19
Npairs	57.56	53.90	66.53	77.54	86.29
Ours	<b>58.39</b>	<b>56.23</b>	<b>68.23</b>	<b>80.06</b>	<b>87.53</b>

(NMI). NMI is defined as the ratio of the mutual information of the clustering and ground truth, and their harmonic mean. Let  $\Omega = \{\omega_1, \omega_2, \dots, \omega_k\}$  be the cluster assignments that are, for example, the result of K-Means clustering. That is,  $\omega_k$  contains the instances assigned to the  $i$ th cluster. Let  $C = \{c_1, c_2, \dots, c_m\}$  be the ground truth classes, where  $c_j$  contains the instances from class  $j$ .

$$NMI(\Omega, C) = 2 \frac{I(\Omega, C)}{H(\Omega) + H(C)} \quad (12)$$

where  $I(\cdot, \cdot)$  and  $H(\cdot)$  denotes mutual information and entropy respectively. Note that NMI is invariant to label permutation which is a desirable property for our evaluation. For more information on clustering quality measurement see [6].

We compare with three state-of-the-art deep metric learning approaches: Triplet Learning with semi-hard negative mining [5], Lifted Structured Embedding [3], and the N-Pairs deep metric loss [4].

We compare the proposed method with all baselines in both clustering and retrieval tasks in Tables 1, 2, and 3. These tables show that lifted structure (LS) [3] and Npair loss (NL) [4], can always improve triplet loss. In particular, N-pair achieves a larger margin in improvement because of the advance in its loss design and batch construction. Compared to previous work, the proposed margin-based discriminate loss consistently achieves better results on all three benchmark datasets. We think the superior performance of Margin-based Discriminate Loss is due to two reasons: (1). It tries to find the optimal embedding space and keep the similar and the dissimilar strictly discriminate. (2). It pulls multiple positive examples together while pushing multiple negative examples away at each update during the training stage. The proposed method involves

two important model parameters: the margin  $m_0$  and the approximate degree  $K$ . The margin  $m_0$  determines to what degree the discrimination would be activated. With the margin  $m_0$  increasing, the network is more difficult to optimize and the performance decrease slowly. We find that when  $K$  is greater than 2, the performance decreases sharply. We select the parameters of our methods via cross-validation on three different datasets. As Fig. 2 shows, choosing  $m_0 = 0.2$  and  $K = 0.8$  for Stanford Cars196 leads to the best performance for the proposed method and our approach is robust to the change of these parameters.

## 6 Conclusion

Triplet loss has been widely used for deep metric learning, even though with somewhat unsatisfactory convergence. In this paper, we firstly propose the strict discrimination concept to seek the optimal embedding space. Based on this concept, we present a novel objective, margin-based discriminate loss, for deep metric learning, which significantly improves upon the triplet loss by pulling multiple positive examples together while pushing multiple negative examples away at a time. The proposed loss function aims to keep the similar and the dissimilar strictly discriminate to find the optimal embedding space at the minimum cost. The proposed method was validated on three benchmark datasets, where the state-of-the-art results validated its efficacy on fine-grained visual object clustering and retrieval.

**Acknowledgement.** This work was supported by the National Natural Science Foundation of China project no. 61772057, in part by Beijing Natural Science Foundation project no. 4162037, and the support funding from State Key Lab. of Software Development Environment.

## References

1. Clarke, F., Ekeland, I.: Nonlinear oscillations and boundary-value problems for Hamiltonian systems. *Arch. Rat. Mech. Anal.* **78**, 315–333 (1982)
2. Suykens, J.A.K., Vandewalle, J.: Least squares support vector machine classifiers. *Neural Process. Lett.* **9**(3), 293–300 (1999)
3. Song, H.O., Xiang, Y., Jegelka, S., et al.: Deep metric learning via lifted structured feature embedding, pp. 4004–4012 (2015)
4. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: *NIPS* (2016)
5. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: a unified embedding for face recognition and clustering. In: *CVPR* (2015)
6. Manning, C.D., Raghavan, P., Schütze, H., et al.: *Introduction to Information Retrieval*, vol. 5. Cambridge University Press, Cambridge (2008)
7. Branson, S., Horn, G.V., Wah, C., Perona, P., Belongie, S.: The ignorant led by the blind: a hybrid human-machine vision system for fine-grained categorization. *Int. J. Comput. Vis.* **108**(1–2), 3–29 (2014)

8. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3D object representations for fine-grained categorization. In: ICCV Workshop on 3D Representation and Recognition (2013)
9. Bai, X., Zhang, H., Zhou, J.: VHR object detection based on structural feature extraction and query expansion. *IEEE Trans. Geosci. Remote Sens.* **52**(10), 6508–6520 (2014)
10. Bai, X., Yang, H., Zhou, J., Ren, P., Cheng, J.: Data-dependent hashing based on p-stable distribution. *IEEE Trans. Image Process.* **23**(12), 5033–5046 (2014)
11. Bai, X., Hancock, E.R., Wilson, R.C.: Graph characteristics from the heat kernel trace. *Pattern Recogn.* **42**(11), 2589–2606 (2009)
12. Bhatia, K., Jain, H., Kar, P., Varma, M., Jain, P.: Sparse local embeddings for extreme multi-label classification. In: NIPS, pp. 730–738 (2015)
13. Bell, S., Bala, K.: Learning visual similarity for product design with convolutional neural networks. *ACM Trans. Graph.* **34**(4), 98:1–98:10 (2015)
14. Li, Y., Su, H., Qi, C.R., Fish, N., Cohen-Or, D., Guibas, L.J.: Joint embeddings of shapes and images via CNN image purification. *ACM Trans. Graph.* **34**(6), 234:1–234:12 (2015)
15. Kiapour, M.H., Han, X., Lazebnik, S., Berg, A.C., Berg, T.L.: Where to buy it: matching street clothing photos in online shops. In: ICCV (2015)
16. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: CVPR (2005)
17. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: DeepFace: closing the gap to human-level performance in face verification. In: CVPR (2014)
18. Choy, C.B., Gwak, J., Savarese, S., Chandraker, M.K.: Universal correspondence network. In: NIPS (2016)
19. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: CVPR (2014)
20. Zhang, X., Zhou, F., Lin, Y., Zhang, S.: Embedding label structures for fine-grained feature representation. In: CVPR (2016)
21. Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J., Ranzato, M., Mikolov, T.: DeViSE: a deep visualesemantic embedding model. In: NIPS (2013)
22. Hsieh, C.-K., Yang, L., Cui, Y., Lin, T.-Y., Belongie, S., Estrin, D.: Collaborative metric learning. In: WWW (2017)
23. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z.: TensorFlow: large-scale machine learning on heterogeneous systems (2015). Software available from [tensorflow.org](http://tensorflow.org)
24. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML, 5 (2015)
25. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. *IJCV* **115**, 211–252 (2015)

# **Dissimilarity Representations and Gaussian Processes**



# Protein Remote Homology Detection Using Dissimilarity-Based Multiple Instance Learning

Antonelli Mensi<sup>1</sup>, Manuele Bicego<sup>1(✉)</sup>, Pietro Lovato<sup>1</sup>, Marco Loog<sup>2</sup>,  
and David M. J. Tax<sup>2</sup>

<sup>1</sup> University of Verona, Verona, Italy  
manuele.bicego@univr.it

<sup>2</sup> Delft University of Technology, Delft, The Netherlands

**Abstract.** A challenging Pattern Recognition problem in Bioinformatics concerns the detection of a functional relation between two proteins even when they show very low sequence similarity – this is the so-called Protein Remote Homology Detection (PRHD) problem. In this paper we propose a novel approach to PRHD, which casts the problem into a Multiple Instance Learning (MIL) framework, which seems very suitable for this context. Experiments on a standard benchmark show very competitive performances, also in comparison with alternative discriminative methods.

**Keywords:** Protein homology · N-grams · Multiple instance learning

## 1 Introduction

The Protein Remote Homology Detection (PRHD) problem represents a relevant bioinformatics problem, widely studied in recent years [1, 12, 14]. It aims at identifying functionally or structurally-related proteins by looking at amino acid sequence similarity – where the term *remote* refers to some very challenging situations where homologous proteins exhibit very low sequence similarity. Many computational approaches have been developed to face this problem – see for example the very recent review published in [1]. In a broad sense, such approaches are divided in three main categories [1]: alignment-based methods, rank-based methods, and discriminative-based methods. Here we focus on this last category, which casts the problem in a binary classification task (homologous/not homologous), and in particular on approaches based on the Support Vector Machines (SVM) classifier – shown to reach top performances in many different benchmarks [6, 14–18, 20].

To apply the SVM, the typical choice is to derive a vectorial representation, so that classic kernels (such as RBF - Radial Basis Function- kernels) can be

---

M. Bicego and P. Lovato were partially supported by the University of Verona through the program “Bando di Ateneo per la Ricerca di Base 2015”.

applied. In this scenario representations based on N-grams (or K-mers<sup>1</sup>) – short subsequences of consecutive symbols – are widely employed [15–18]. The well known Bag of Words representation is an example of such characterization [7, 15, 17, 18]. Here a vectorial representation is extracted consisting of the number of times the dictionary N-grams appear in the sequence. Although this leads to excellent results, the main problem of this class of approaches is that  $N$  (i.e. the length of the subsequence) is forced to remain small (such as 3). For longer N-grams, the representation becomes too large (leading to the curse of dimensionality) and too sparse (with too many zeros), thus creating problems to the SVM [4]. Actually, due to the limited length, we can not fully exploit the biological information present in longer sequences. An alternative is to devise methods which directly compute kernels on the basis of long K-mers, avoiding the explicit computation of the representation. One notable example is [11], where authors propose a K-mer based string kernel approach. In their work they showed that the best performances are obtained with K-mers of length 5.

In this paper we propose a novel approach to PRHD, which derives a novel vectorial representation for SVM-based discriminative techniques. The approach is based on the paradigm of Multiple Instance Learning (MIL – [5]), an extension of supervised learning where class labels are associated with sets (bags) of feature vectors (instances) rather than with individual feature vectors. This paradigm, whose usefulness has been shown in many different contexts [2, 8], has not yet been investigated in the Protein Remote Homology Detection scenario. Here we cast the PRHD problem in a MIL framework by interpreting protein sequences as bags that contain fragments of a certain length  $k$  (the instances). The classification problem is solved using a recent MIL approach based on dissimilarities between instances [3]. The MIL scenario, and in particular the dissimilarity-based approach of [3], seems to be very suitable for the PRHD problem for different reasons. First, the MIL paradigm assumes that the label of the whole bag is determined by only a small set of relevant instances [5]. This assumption is reasonable in PRHD, where the homology between two proteins is linked to the presence of a small set of highly informative fragments (such as ligand sites). Second, it does not impose any limit to the length of the K-mers, so that also biologically meaningful longer fragments can be included in the analysis. Third, the approach of [3] relies on the computation of distances between instances, which in the PRHD case can be easily defined via meaningful sequence alignment methods.

The proposed approach, presented in some different variants, has been tested using standard benchmarks based on the SCOP 1.53 dataset [14]. The results confirm the suitability of the proposed approach, also in comparison with alternative discriminative methods.

---

<sup>1</sup> Along the text we will refer equivalently to K-mers or N-grams.

## 2 General and Dissimilarity-Based MIL

In this section we introduce the general multiple instance learning paradigm, together with the approach presented in [3] that we used. Multiple Instance Learning (MIL – [5]) is concerned with problems where the objects originally are not represented by a single feature vector, but by a so-called bag. A bag is basically a *set* of feature vectors, the latter of which are also referred to as instances in this context. As opposed to the standard classification setting, a label is then assigned to the whole bag and not the individual feature vectors. This can make classification quite difficult. The basic assumption behind MIL is that a positive label of a bag indicates the presence of (at least) a positive instance inside the bag – we will see that this assumption is very suitable for our context.

Many different approaches have been proposed to solve MIL problems [2, 8], here we summarize the methods proposed in [3]. These methods are based on the dissimilarity-based paradigm for classification [19], a paradigm where each object is represented by a vector of dissimilarities with respect to a set of reference objects (called prototypes). In the same spirit, in the approach of [3] each bag is encoded into a vectorial representation based on the distances between the instances of the bag and the instances of a set of prototypes.

More in detail, we are given  $N$  bags to encode and a set of  $L$  prototypes. The choice of these prototypes is crucial, but in the basic version they can also be the whole training set. Given prototype  $P_j$  containing  $m$  instances,  $P_j = \{x_{j1}, \dots, x_{jm}\}$ , we represent a bag  $B_i = \{x_{i1}, \dots, x_{in}\}$  with  $n$  instances, by some signature extracted from the pairwise distances between all the instances of  $B_i$  and those of the prototype bag  $P_j$ . Different features can be extracted from the resulting  $n \times m$  dissimilarity matrix.

1.  $d_{bag}$  feature. This feature is a scalar, and represents the average of the minimum distances between each fragment of the bag and all the fragments of the prototype.

$$d_{bag}(B_i, P_j) = \frac{1}{|B_i|} \sum_{k=1}^{|B_i|} \min_l d(x_{ik}, x_{jl})$$

where  $d(x_{ik}, x_{jl})$  represents a distance between instances of the bag.

2.  $d_{inst}$  feature. This is a vector of length  $m$ , where each component represents the minimum distance between each fragment of the prototype and all fragments of the bag.

$$d_{inst}(B_i, P_j) = \left[ \min_k d(x_{ik}, x_{j1}), \dots, \min_k d(x_{ik}, x_{jm}) \right]$$

In the first two MIL schemes, which are called  $D_{bag}$  and  $D_{inst}$ , each bag is represented by concatenating all the  $d_{bag}$  and  $d_{inst}$  features computed with respect to all prototypes, i.e.  $D_{bag}(B_i) = [d_{bag}(B_i, P_1), d_{bag}(B_i, P_2), \dots, d_{bag}(B_i, P_L)]$  and  $D_{inst}(B_i) = [d_{inst}(B_i, P_1), d_{inst}(B_i, P_2), \dots, d_{inst}(B_i, P_L)]$ .



These representations may have some limitations:  $D_{bag}$  may hide the most informative dissimilarities, since it is an average over all distances, not considering that only few instances are relevant. The  $D_{inst}$  method, on the contrary, considers all these dissimilarities, but the process of selection can be time consuming. Furthermore it may suffer from the curse of dimensionality. To overcome these possible limitations, the authors in [3] proposed a variant which exploits the combining classifier paradigm. The method, which we call the “ensemble” approach, is based on considering each prototype as a single subspace where a classifier is trained. Similarly to the  $D_{inst}$  method, each direction of the subspace represents the minimum distance between each instance of the prototype and all instances of the bag. The dimensionality of this subspace is therefore the number of instances of the prototype. Given  $L$  prototypes, we built  $L$  different representations, training  $L$  different classifiers. The final classifier is then found by aggregating the results of the  $L$  different classifiers via a combining function (in this sense it is an ensemble approach) – for further details please refer to [3].

### 3 MIL Solution to the PRHD Problem

In our proposed approach we first cast the PRHD problem into a MIL formulation, i.e. we define bags, instances and labels. This is done in a reasonable and straightforward way: (i) each protein sequence is a bag, i.e. a collection of N-grams (instances); (ii) the fragments (N-grams) composing the protein sequence are considered the instances; (iii) finally, the label, which is attached to the set of instances, is the label of the sequence. Please note that MIL represents a natural representation for the PRHD problem: proteins typically contain a small set of meaningful fragments, which are crucial to determine the 3D structure (e.g. binding sites) and thus the function (namely the label). Clearly, the fragments can be extracted from the sequence in many different ways (random sampling, exhaustive list, and so on). Here we adopt a very simple scheme: from each sequence of length  $n$ , fragments of a fixed length  $k$  are extracted with overlap  $k - 1$ . Each bag  $B_i$  will therefore have  $n - k + 1$  instances. Once cast into a MIL formulation, the PRHD problem is then input to the dissimilarity-based approach presented in the previous section. In particular, a set of prototypes  $\mathcal{P} = \{P_1 \dots P_L\}$  is chosen as a subset of the training set  $\mathcal{T}$ . Given a prototype  $P_j$ , for each sequence  $S_i$  we compute a dissimilarity matrix between all fragments of  $P_j$  and all the fragments of  $S_i$  (i.e. the bag  $B_i$ ). As described in the previous section, from this matrix we then derive two different representations: a scalar ( $d_{bag}$ ) or a set of values ( $d_{inst}$ ). In the basic formulation, the dissimilarity matrices are extracted for all prototypes and concatenated to obtain the final representation of our sequence. The proposed representation can now be fed to the SVM classifier. Alternatively, the ensemble method described in the previous section can be used: the classifier is trained on  $d_{inst}$  of a single prototype, called a subspace, and then the obtained scores are combined together to obtain the final results via an ensemble classifier. Summarizing, we have three different MIL schemes: one using ( $D_{bag}$ ), one using ( $D_{inst}$ ), and the last using the ensemble approach ( $D_{ens}$ ).

One crucial aspect of this class of approaches is the choice of the prototypes. First, the number of prototypes has to be chosen. Next, it is crucial to define the strategy with which they are chosen. Here we studied three different options:

- (i) **Random choice of sequences:** the prototypes are randomly selected protein sequences of the training set.
- (ii) **Informed choice of sequences:** the prototypes are chosen exploiting some a priori knowledge on the training set.
- (iii) **Random fragments:** here the prototypes are not anymore objects of the training set (i.e. whole sequences), but they are built using random fragments extracted from sequences. After deciding on the number of fragments that should compose each prototype, we randomly select those fragments from the whole set of bags. Note that our proposed scheme allows to exploit long K-mers without increasing in a significant way the dimensionality. In fact, the dissimilarity matrix between bag's instances, which is at the basis of our scheme, does not depend from the length of the K-mers, but only the the number. This permits to exploit longer fragments with respect to classic N-grams methods, which may contain more important biological information, such as that related to folding.

## 4 Experiments

The proposed approach has been tested on the standard benchmark dataset<sup>2</sup>, based on the SCOP 1.53 [14]. Even if quite old and not complete, this represents a standard dataset for protein remote homology detection, permitting to compare most of the methods introduced in this field [6, 14–18, 20]. Following the standard protocol introduced in [14], the PRHD problem has been cast in a set of 54 binary classification problems, each one involving a specific protein family. As done in some recent studies [15–17], before extracting N-grams we re-wrote each protein sequence using information extracted from the corresponding profile, determined by following the recent [16], which employed a public implementation of the PsiFreq program<sup>3</sup>.

Once determined, the MIL representations are then employed to train a SVM classifier. As done in many previous works [7, 15–18, 20], we used the public GIST implementation<sup>4</sup>, setting the kernel type to radial basis, and keeping the remaining parameters to their default values. Detection accuracies are measured using the ROC50 score [9]. This score, specifically designed for the PRHD context, improves the classic Area under the ROC curve. In particular, it represents the area under the ROC50 curve (with a value ranging from 0 to 1), which plots true positives as a function of false positives – up to the first 50 false positives. A score of 1 indicates perfect separation of positives from negatives, whereas a score of 0 indicates that none of the top 50 sequences selected by the algorithm were positives [13].

<sup>2</sup> Available at <http://noble.gs.washington.edu/proj/svm-pairwise/>.

<sup>3</sup> Available at <http://bioinformatics.hitsz.edu.cn/main/~binliu/remote>.

<sup>4</sup> Downloadable from <http://www.chibi.ubc.ca/gist/> [14].

For the proposed approach, we repeated the experiment for  $k = \{2, 3, 4, 5, 6, 9, 12\}$ . The distance between the K-mers was computed using the classic Jukes-Cantor distance, based on the Hamming distance. Please note that this is a basic distance between sequences, which does not imply any alignment. It can be expected that performances may improve even more when more advanced sequence comparison methods are used, for instance methods that allow for the comparison of K-mers of different lengths. We tested different variants of the proposed approach, trying to cover the most interesting combinations of the basic scheme ( $(D_{bag})$ ,  $(D_{inst})$ , and  $(D_{ens})$ ) and the way prototypes are chosen. For all variants we investigated two possible options, which derive from the fact that the benchmark contains 54 classification problems. In particular, in the first version (called SfA – Same for All) the prototypes were kept identical among all 54 problems. In the second version (called DfA - Different for All) a different set of prototypes is used for each family. In particular the following variants have been investigated:

- (i)  **$D_{bag}$ -Info.** In this variant, we used the  $D_{bag}$  information to build the representation, choosing the prototypes in an informed way. In the SfA version, we used 54 prototypes, equal for all families: each prototype is the most central sequence of the positive training set of each family, that is the one with lowest distance to all other sequences. In the DfA version, for each family we used as prototypes all the sequences in the positive part of training set.
- (ii)  **$D_{inst}$ -Info.** In this variant we used the  $D_{inst}$  information to build the representation. Due to the high dimensionality of this representation, we choose to employ a single prototype, chosen in an informed way. In particular, in the SfA version, the prototype was chosen as the most central sequence among all positive training sequences of the 54 families. In the DfA version, for each family the prototype was chosen as the most central sequence among the positive training sequences of the considered family.
- (iii)  **$D_{inst}$ -RndFrag.** In this variant we used again the  $D_{inst}$  information to build the representation, employing again one prototype. However the prototype was chosen using random fragments. In the SfA version, the fragments are extracted from the set composed by the fragments of all the positive training sequences of all families. The cardinality of the prototype P is the ratio between the total number of fragments of the just mentioned bag and the total number of positive training sequences. In the DfA version, for each family the random fragments are chosen among the set composed by the fragments of all the positive training sequences of the considered family. The cardinality of each prototype P is the ratio between the total number of fragments of the just mentioned bag and the number of positive training sequences.
- (iv)  **$D_{ens}$ -RndSeq-Mean.** In this variant we used the ensemble MIL scheme to build the representation, using random sequences as prototypes. In particular, in the SfA version, we randomly chose 10 prototypes from the set of all positive training sequences of the 54 problems. Then we extract the

$D_{inst}$  representation for each prototype, training a different SVM for each of them. Once computed the SVM scores, a “mean” combiner function is used to get the final score (i.e. the mean of all scores). In the DfA version, the 10 prototypes were different for each classification problem. In particular, for each family we selected 10 prototypes from the set of positive training sequences of that family. A study on the performances by using a different number of prototypes is reported later.

- (v)  **$D_{ens}$ -RndSeq-Max**. This is identical to the  **$D_{ens}$ -RndSeq-Mean** except that the combiner was a “max” combiner (i.e. the max among the scores).
- (vi)  **$D_{ens}$ -RndFrag-Mean**. This variant is similar to  **$D_{ens}$ -RndSeq-Mean**, except that the prototypes are built using Random Fragments. Prototypes, for both SfA and DfA versions are determined as described in the  **$D_{inst}$ -RndFrag** variant. In this version we used the “mean” combiner.
- (vii)  **$D_{ens}$ -RndFrag-Max**. This is identical to the  **$D_{ens}$ -RndFrag-Mean** except that we used the “Max” combiner.

For each experiment we selected the best result among the different lengths of N-grams (which can be reasonably different depending on the specific family addressed). A further analysis on the preferred length has been reported later in the section. ROC50 values, averaged over the 54 families, are reported in Table 1, for the different variants. From the table we make different observations. First, it is interesting to note that the most basic variant of our scheme, namely the  **$D_{bag}$ -Info**, is performing very well, at the same level of the most complicated variants. This suggests that the extracted information, even in its basic form, is already very informative. Second, it seems evident that choosing the same set of prototypes for all families permits to reach better performances in almost all cases. Actually we are convinced that the crucial point is not that the prototypes are the same for all classification problem (each classification problem is solved independently), but rather that this set is chosen among the whole set of sequences rather than the single training set of a given family. This permits to have a more variable set of prototypes which permits to get a richer representation. Interestingly, the informed choice of the prototypes does not improve in a substantial way the performances. As a final observation, it is important

**Table 1.** ROC50 accuracies of the different variants of the proposed approach.

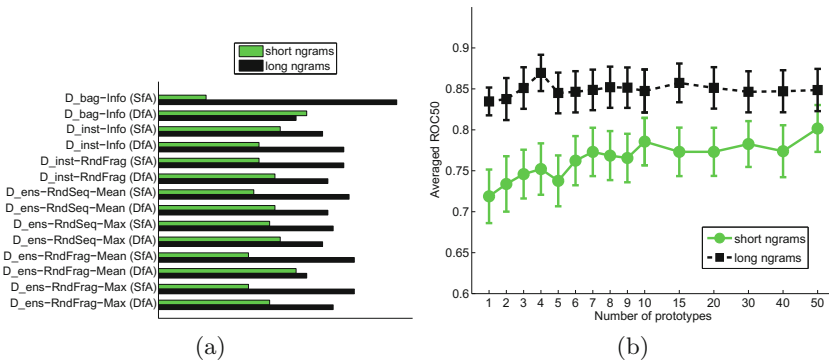
Variant	MIL scheme	Prot. Sel.	ROC50 (SfA)	ROC50 (DfA)
<b><math>D_{bag}</math>-Info</b>	$D_{bag}$	Informed	0.863	0.711
<b><math>D_{inst}</math>-Info</b>	$D_{inst}$	Informed	0.820	0.781
<b><math>D_{inst}</math>-RndFrag</b>	$D_{inst}$	Rand Frag	0.867	0.862
<b><math>D_{ens}</math>-RndSeq-Mean</b>	$D_{ens}$	Rand Seq	0.878	0.792
<b><math>D_{ens}</math>-RndSeq-Max</b>	$D_{ens}$	Rand Seq	0.819	0.781
<b><math>D_{ens}</math>-RndFrag-Mean</b>	$D_{ens}$	Rand Frag	0.882	0.847
<b><math>D_{ens}</math>-RndFrag-Max</b>	$D_{ens}$	Rand Frag	0.837	0.878

**Table 2.** Results of the variant  $D_{ens}$ -RndFrag-Mean (SfA) with varying number of prototypes.

Nr. prototypes	1	2	3	4	5	7	10	15	20	30	40	50
ROC 50	0.867	0.872	0.886	0.892	0.880	0.882	0.882	0.874	0.879	0.868	0.870	0.880

to note that when combining the classifiers in the  $D_{ens}$  class of approaches the best result is obtained with the mean rule (in line with other studies in classifiers combination [10]).

In order to see how critical the number of prototypes  $L$  is, we performed another set of experiments using the best performing technique, i.e. the variant  $D_{ens}$ -RndFrag-Mean (SfA). We varied the number of prototypes from 1 to 50, and the corresponding accuracies are reported in Table 2. It appears that performances do not vary too much when more than 3 prototypes are used. This suggests that the approach is robust against variations in  $L$ , provided that this number exceeds a minimum (3 in this case). Another interesting aspect to be analysed concerns the length of the K-mers. As already mentioned, in our experiments we computed results by varying the length  $k$  of the fragments, selecting, for each family, the length leading to the best accuracy. It seems interesting to observe the distribution of such best  $k$ , in order to discover if the MIL approach prefers short or long N-grams. To do that, for each variant, we count how many times the best result is obtained with *short N-grams* (N-grams of length 2 or 3) or with *long N-grams* (N larger than 3). Such analysis is reported in Fig. 1(a). In all cases except the  $D_{bag}$ -Info (DfA) variant, longer fragments give better results. Furthermore, in Fig. 1(b) the accuracies obtained by  $D_{ens}$ -RndFrag-Mean (SfA) are shown for an increasing number of prototypes (results of Table 2), divided in two cases: method with *short N-grams* and

**Fig. 1.** Analysis of preferred N-gram length: (a) the distribution of the best length over all approaches and (b) the ROC50 performance as a function of the number of prototypes.

method with *long N-grams*. The results with *long N-grams* are better and seem to be more independent from the number of prototypes (whereas with *short N-grams* there seems to be an increasing behaviour). All these findings confirm our intuition that exploiting longer fragments can be beneficial for facing the Protein Remote Homology Detection problem.

#### 4.1 Comparison with the State of the Art

In Table 3 we compared the proposed scheme with alternative approaches present in the literature. The SCOP 1.53 dataset, even if being old, has been widely used as benchmark for many different approaches. We reported in the table comparative results taken from the very recent [17], which are related to both Bag of Words approaches as well as more complicated alternatives. We can see that the proposed approach is very competitive, well comparing with alternatives. In particular, the proposed approach is better than almost all methods presented in the table, with the exception of the very complex Soft PLSA approach [17]: this recent method, however, starts from a larger set of information – the complete profile of each protein together with evolutionary probabilities – whereas our approach only uses the most probable profile (for more information, interested readers are referred to [17]).

**Table 3.** Comparison with state of the art. For the proposed approach we reported the best obtained result, i.e. the result for **D<sub>ens</sub>-RndFrag-Mean** (SfA) with 4 prototypes – see Table 2.

<i>N-grams based approaches</i>			<i>Other approaches</i>		
<b>Method</b>	<b>Year</b>	<b>ROC50</b>	<b>Method</b>	<b>Year</b>	<b>ROC50</b>
BoW-row-2gram	2017	0.772 [17]	SVM-pairwise	2014	0.787 [16]
Soft BoW	2017	0.844 [17]	SVM-LA	2014	0.752 [16]
Soft PLSA	2017	0.917 [17]	HHSearch	2017	0.801 [17]
SVM-N-gram	2014	0.589 [16]	Profile (5,7.5)	2005	0.796 [11]
SVM-N-gram-LSA	2008	0.628 [15]	PSI-BLAST	2007	0.330 [6]
SVM-Top-N-gram (n = 2)	2008	0.713 [15]	SVM-Bprofile-LSA	2007	0.698 [6]
SVM-Top-N-gram- combine	2008	0.763 [15]	SVM-Pattern-LSA	2008	0.626 [15]
SVM-N-gram-p1	2014	0.726 [16]	SVM-Motif-LSA	2008	0.628 [15]
SVM-N-gram-KTA	2014	0.731 [16]	SVM-LA-p1	2014	0.888 [16]

ROC50 of the proposed approach: 0.892

## 5 Conclusions

In this paper we presented a Multiple Instance Learning approach for Protein Remote Homology detection. The proposed scheme casts the PRHD problem into the MIL paradigm by considering protein sequences as bags of N-grams, i.e. short fragments of the sequence. A dissimilarity-based approach is then used to face the MIL problem, based on the matrix of pairwise distances of fragments of a given protein and fragments of a set of prototypes. An empirical evaluation on standard datasets confirms the suitability of the proposed framework. Future directions include analysis of richer dissimilarities as well as the selection of biologically relevant prototypes (e.g. binding sites).

## References

1. Chen, J., Guo, M., Wang, X., Liu, B.: A comprehensive review and comparison of different computational methods for protein remote homology detection. *Brief. Bioinf.* **19**, 1–14 (2016)
2. Chen, Y., Bi, J., Wang, J.Z.: MILES: multiple-instance learning via embedded instance selection. *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(12), 1931–1947 (2006)
3. Cheplygina, V., Tax, D., Loog, M.: Dissimilarity-based ensembles for multiple instance learning. *IEEE Trans. Neural Netw. Learn. Syst.* **27**(6), 1379–1391 (2016)
4. Cucci, A., Lovato, P., Bicego, M.: Enriched bag of words for protein remote homology detection. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) *S+SSPR 2016*. LNCS, vol. 10029, pp. 463–473. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_41](https://doi.org/10.1007/978-3-319-49055-7_41)
5. Dietterich, T., Lathrop, R., Lozano-Pérez, T.: Solving the multiple instance problem with axis-parallel rectangles. *Artif. Intell.* **89**(1–2), 31–71 (1997)
6. Dong, Q., Lin, L., Wang, X.: Protein remote homology detection based on binary profiles. In: Hochreiter, S., Wagner, R. (eds.) *BIRD 2007*. LNCS, vol. 4414, pp. 212–223. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-71233-6\\_17](https://doi.org/10.1007/978-3-540-71233-6_17)
7. Dong, Q., Wang, X., Lin, L.: Application of latent semantic analysis to protein remote homology detection. *Bioinformatics* **22**(3), 285–290 (2006)
8. Fung, G., Dundar, M., Krishnapuram, B., Rao, R.: Multiple instance learning for computer aided diagnosis. *Proc. Adv. Neural Inf. Process. Syst.* **19**, 425–432 (2007)
9. Gribskov, M., Robinson, N.: Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. *Comput. Chem.* **20**(1), 25–33 (1996)
10. Kittler, J., Hatef, M., Duin, R.P., Matas, J.: On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.* **20**(3), 226–239 (1998)
11. Kuang, R., Wang, K., Wang, K., Siddiqi, M., Freund, Y., Leslie, C.: Profile-based string kernels for remote homology detection and motif extraction. *J. Bioinf. Comput. Biol.* **3**(03), 527–550 (2005)
12. Kuksa, P.P., Pavlovic, V.: Efficient evaluation of large sequence kernels. In: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 759–767. ACM (2012)
13. Leslie, C., Eskin, E., Noble, W.: The spectrum kernel: a string kernel for SVM protein classification. In: *PSB*, pp. 566–575 (2002)

14. Liao, L., Noble, W.: Combining pairwise sequence similarity and support vector machines for detecting remote protein evolutionary and structural relationships. *J. Comput. Biol.* **10**(6), 857–868 (2003)
15. Liu, B., Wang, X., Lin, L., Dong, Q., Wang, X.: A discriminative method for protein remote homology detection and fold recognition combining top-n-grams and latent semantic analysis. *BMC Bioinf.* **9**(1), 510 (2008). <https://doi.org/10.1186/1471-2105-9-510>
16. Liu, B., et al.: Combining evolutionary information extracted from frequency profiles with sequence-based kernels for protein remote homology detection. *Bioinformatics* **30**(4), 472–479 (2014)
17. Lovato, P., Cristani, M., Bicego, M.: Soft Ngram representation and modeling for protein remote homology detection. *IEEE/ACM Trans. Comput. Biol. Bioinf.* **14**(6), 1482–1488 (2017)
18. Lovato, P., Giorgetti, A., Bicego, M.: A multimodal approach for protein remote homology detection. *IEEE/ACM Trans. Comput. Biol. Bioinf. (TCBB)* **12**(5), 1193–1198 (2015)
19. Pekalska, E., Duin, R.P.W.: *The Dissimilarity Representation for Pattern Recognition: Foundations and Applications, Machine Perception and Artificial Intelligence*, vol. 64. World Scientific, Singapore (2005)
20. Rangwala, H., Karypis, G.: Profile-based direct kernels for remote homology detection and fold recognition. *Bioinformatics* **21**(23), 4239–4247 (2005)





# Local Binary Patterns Based on Subspace Representation of Image Patch for Face Recognition

Xin Zong<sup>(✉)</sup>

Graduate School of Systems and Information Engineering,  
University of Tsukuba, Tsukuba, Japan  
zongxiaoxin@gmail.com

**Abstract.** In this paper, we propose a new local descriptor named as PCA-LBP for face recognition. In contrast to classical LBP methods, which compare pixels about single value of intensity, our proposed method considers that comparison among image patches about their multi-dimensional subspace representations. Such a representation of a given image patch can be defined as a set of coordinates by its projection into a subspace, whose basis vectors are learned in selective facial image patches of the training set by Principal Component Analysis. Based on that, PCA-LBP descriptor can be computed by applying several LBP operators between the central image patch and its 8 neighbors considering their representations along each discretized subspace basis. In addition, we propose PCA-CoALBP by introducing co-occurrence of adjacent patterns, aiming to incorporate more spatial information. The effectiveness of our proposed two methods is accessed through evaluation experiments on two public face databases.

**Keywords:** Local Binary Pattern · Principal Component Analysis  
Subspace Representation · Image Patch · One Sample per Person

## 1 Introduction

“One Sample per Person” problem is a challenging topic in face recognition due to the limited representative of reference sample. The goal is to identify a person from the database later in time in any different and unpredictable poses, lighting, etc. from just one image [14]. For attacking that problem, many local feature methods are applied and achieve good performance due to their computational simplicity and robustness to occlusion and illumination. One of the most well-known is Local Binary Pattern (LBP). Although it is firstly introduced to describe texture, which could be characterized by a nonuniform distribution of intensity or colors [4], it is then extensively used in face recognition motivated by the fact that face can be seen as a composition of micro-patterns which are well described by such operator [1].

However, designing a robust local descriptor is not an easy job. And most hand-crafted features cannot be simply adopted to new conditions [2,6]. In

recent years, many learned-based methods are proposed for designing better local descriptor. For example, PCANet [3] learns its binary descriptor by binarizing the convolution results of local image patch with several learned linear filters. Other methods such as L2-Net [16], which attempt to use CNN based methods, are proposed to construct more robust descriptors for high matching performance.

While for face recognition, it can be difficult for these learned descriptors to capture macro-structures due to their well-but-micro representation limited in local patch. That limitation gives rise to our idea of PCA-LBP, aiming to encode macro facial patterns by applying LBP operators among image patches. Since classical LBP methods successfully capture micro-patterns in the level of pixel, which is the smallest addressable element, it can be natural to consider that a macro-pattern is possible to encode by applying LBP in the level of image patch, which is a container of pixels in larger form.

To implement LBP in the level of image patch, there can be two main problems. The first is to find an efficient representation of facial image patch. Many possible methods have been investigated for data characterization, one of the most simple-but-efficient is Principal Component Analysis. The PCA allows us to characterize an image patch by its projection on a linear subspace. However, such a subspace representation can be multi-dimensional, thus leading to the second problem about how classical LBP can be implemented for comparison of multi-dim values. Standard LBP compares pixels' intensity, which is virtually a single value, while the subspace representation can be multi dimensional. To address that problem, we introduce a set of LBP operators instead of a single one. And each LBP operator is discretely implemented between the object image patch and its 8 neighbors considering their representations along the corresponding subspace basis.

This concept of patch representation by PCA and patch comparison by several LBPs is at the heart of our proposed method, thus we name it as PCA-LBP. Moreover, our proposed method can be generically described as a hybrid model of original LBP in pixel level with learned descriptor in image patch level. This characteristic makes it possible to be flexibly transferred with other LBP methods. Therefore, PCA-CoALBP, which considers co-occurrence of adjacent LBPs, is also proposed. To confirm the robustness of our proposed two descriptors for face representation, we assess them for attacking one sample per person problem in two public face databases: Extended Yale Face B Database and AR Face Database.

The contributions of this paper are listed as follows:

- We review PCANet in the new perspectives from binary descriptor and image patch subspace, which is critical in developing our proposed methods.
- We propose two new local descriptors: PCA-LBP and PCA-CoALBP, aiming to explore a hybrid framework, which combines the classical LBP in pixel level with the learned descriptor in image patch level.
- We confirm the effectiveness of our proposed methods for face recognition in two benchmark face databases.

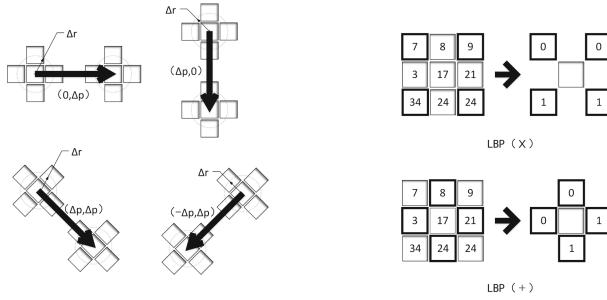


Fig. 1. Configuration of CoALBP

## 2 Related Work

In this section, we review two related research: (1) local binary pattern, and (2) PCANet.

### 2.1 LBP and CoALBP

LBP computes a bit string by comparing intensity in center pixel with its 8 neighboring pixels. In [12], the definition of LBP is mathematically given as follows:

$$LBP_R(x) = \sum_{i=0}^7 \text{sign}(I(x_i) - I(x))2^i \tag{1}$$

Where R defines the distance of center pixel  $x$  to its neighborhood  $x_i$ . Recent studies show that encoding co-occurrences of local binary patterns can significantly improve the performance [13]. In [11], a new descriptor based on Co-occurrence of Adjacent Local Binary Patterns (CoALBP) is proposed and achieve good performance both in texture classification and face recognition. The core idea of it is to introduce a statistical count about the frequency of adjacent LBP pairs in a fixed spatial distance. Figure 1 shows that CoALBP computes frequency of LBP pairs in 4 directions with a configured  $\Delta r$  (scale of LBP radius) and  $\Delta p$  (interval of LBP pairs). In addition, as can be seen, CoALBP considers two sparse LBP configuration - LBP(+) and LBP(x), aiming to reduce computational time.

### 2.2 PCANet

Given an image patch  $x$ , its descriptor by one layer PCANet (PCANet-1) may be defined as a string of binary code. Elements in that binary string can be computed by thresholding the convolution results of its local patch with several PCA filters. While in the perspective of image patch subspace, the binary descriptor of  $x$  can be described by thresholding its subspace representation, which is computed by its projection into an image patch subspace. And the basis vectors

of that subspace are virtually the pre-learned PCA filters with vector notation. The final binary descriptor of image patch  $x$  is obtained by thresholding each element in its subspace representation by comparison with zero.

In our study, we do not utilize that binary descriptor. Instead, we only introduce the idea of finding subspace representation of image patch via Principle Component Analysis into our proposed methods. In addition, our interpretation of PCANet is inspired by the pioneer research of BSIF [8], which illustrates its binary descriptor from the perspective of image patch subspace. However, the subspace basis in BSIF is generated by Independent Component Analysis. Therefore, it is not the same as PCANet.

### 3 Proposed Method

In this section, we illustrate the core idea of PCA-LBP in constructing local descriptor and extracting image histogram feature. Note that for PCA-CoALBP, the only difference is to apply several CoALBP operators instead of LBP operators in the stage of encoding.

#### 3.1 Local Descriptor

Figure 2 shows the process flow of constructing a PCA-LBP descriptor for a given image patch  $x$ . As can be seen, its 8 neighbors  $\{x_i\}_{i=0}^7$  are taken into consideration for encoding marco-pattern. Overall, there are three stages in the processing. The initial stage is to apply Principal Component Analysis to find the subspace representation  $\{S_j(x)\}_{j=1}^N$  of image patch  $x$  as shown in (2).

$$\{S_j(x)\}_{j=1}^N = \{W_j^T \cdot \tilde{x}\}_{j=1}^N \quad (2)$$

Where  $W_j$  defines the  $j$ th subspace basis,  $N$  indicates the dimension of pre-learned subspace and  $\tilde{x}$  denotes vectorized image patch  $x$  with its DC component removed. DC component refers to mean gray-value of the pixels in that image patch [7]. And each  $S_j(x)$  is virtually the projected length of  $\tilde{x}$  along the corresponding  $j$ th subspace basis  $W_j$ . In addition,  $\{W_j\}_{j=1}^B$  can be constructed by retaining first  $N$  th principal component in a training set of image patches. Next, such subspace representations of  $x$  and its 8 neighbors are encoded by several LBP operators. Specifically, each LBP operator compares the subspace representation  $S_j(x)$  of image patch  $x$  along corresponding subspace basis  $W_j$  with that of its 8 neighbors. The stage is then followed by concatenating the encoding result of those LBP operators. Finally, the PCA-LBP descriptor of image patch is obtained and can be mathematically defined as  $\{P_j(x)\}_{j=1}^N$  in (3).

$$PCA - LBP_{R,N}(x) = \{P_j(x)\}_{j=1}^N = \left\{ \sum_{i=0}^7 \text{sign}(S_j(x_i)) - S_j(x) 2^i \right\}_{j=1}^N \quad (3)$$

Where  $R$  defines the radius distance between image patch  $x$  and its neighbors  $\{x_i\}_{i=0}^7$ ,  $\text{sign}$  functions as the LBP thersholding and  $N$  indicates the number of LBP operators.

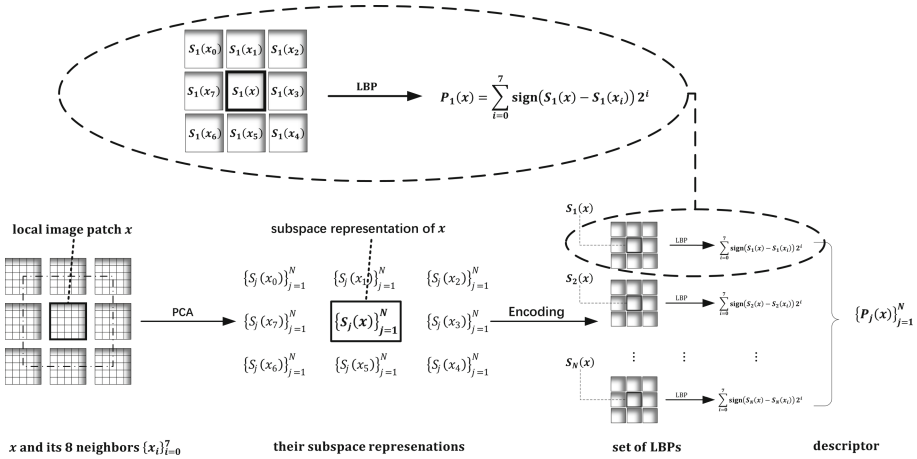


Fig. 2. PCA-LBP descriptor of an image patch

### 3.2 Image Histogram Feature

Figure 3 shows the PCA-LBP histogram feature of an input image. Given an input image  $X$  of size  $H \times W$  pixels, its histogram representation by PCA-LBP can be mathematically defined as  $F(X)$  in (4).

$$F(X) = [\text{hist}(X_1); \text{hist}(X_2); \dots; \text{hist}(X_N)] \quad (4)$$

$F(X)$  can be described as a concatenation of block-wise histograms of several relabelled images  $\{X_j\}_{j=1}^N$ .  $N$  indicates length of PCA-LBP descriptor and  $\{X_j\}_{j=1}^N$  denotes several shift-equivalent images of  $X$  by PCA-LBP processing.

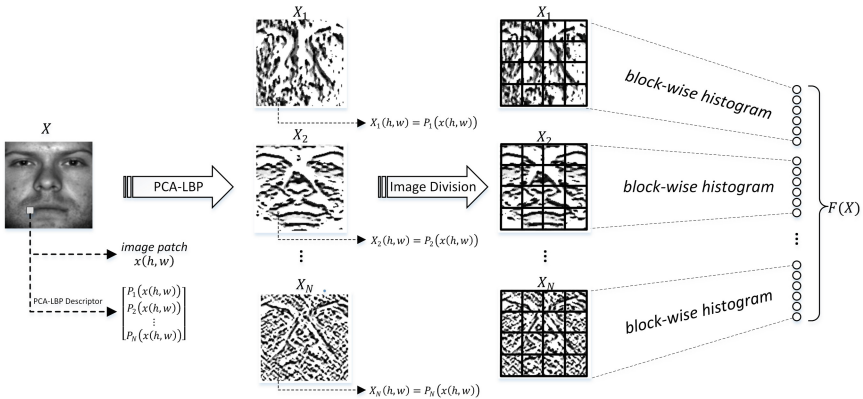


Fig. 3. PCA-LBP histogram feature of an input image



**Fig. 4.** Examples in Extended Yale Face B Database

In addition, as can be seen, given a patch  $x(h, w)$  in input image  $X$ , its corresponding value  $X_j(h, w)$  in relabeled image  $X_j$  can be computed as follows:

$$X_j(h, w) = P_j(x(h, w)) \quad (5)$$

Where  $P_j(x(h, w))$  indicates the  $j$ th element value in the PCA-LBP descriptor of  $x(h, w)$ .

## 4 Experiments and Considerations

In this section, we illustrate details of our experiments in two public face databases for attacking one sample per person problem.

### 4.1 Face Recognition in Extended Yale Face B Database

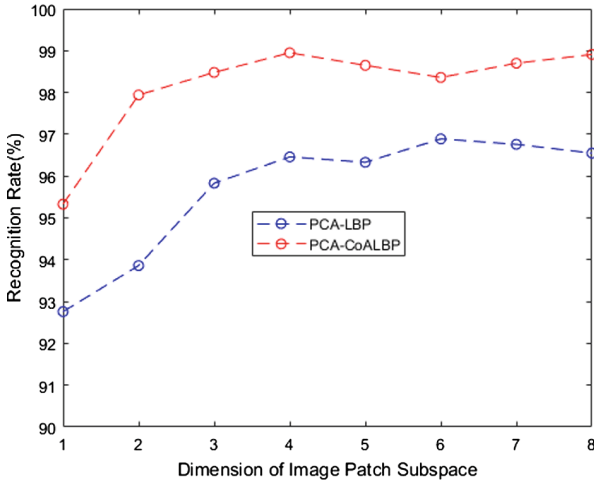
In this experiment, we focus on attacking one sample per person problem under difficult lighting conditions.

**Database.** Extended Yale Face B Database contains face images of 38 subjects of 9 poses under 64 illuminations [9]. We use 2414 frontal-face images in our experiment. Figure 4 shows an example of frontal facial images of one subject under variable lighting.

**Setup.** In our experiment, all facial images are resized to  $126 \times 126$  pixels and divided into  $7 \times 7$  non-overlapped subregions. 38 frontal-lighting images (one sample per person) are selected as reference images. The rest 2376 images are used for testing. In addition, 114 images (3 for each sample) are synthesized by artificially adding Gaussian noise and slight rotation into original reference images. Those synthesized images and reference images are transformed into image patches for learning principal components. And the key parameters involved in our proposed two methods are listed as follows:

- size of image patch:  $k$
- scale of LBP radius:  $\Delta r$
- interval of LBP pair:  $\Delta p$
- configuration on LBP: config (x or +)
- dimension of image patch subspace:  $N$ .

PCA-CoALBP considers all parameters while PCA-LBP considers three of them:  $\Delta r, N$  and  $k$ . In this experiment, patch size  $k$  is empirically set as  $5 \times 5$  pixels. And 1-NN method based on L1 distance is used for classification.



**Fig. 5.** Impact of dimension selection

**Parameter Impact.** Since there are several parameters included in our methods, a strategy to help us find the best parameter set is to utilize original LBP methods. The best selection of parameters in original LBP and CoALBP helps to define the range of those parameters in our methods such as  $\Delta r$  and  $\Delta p$ . Therefore, the core parameter to be investigated is  $N$  - dimension of image patch subspace. Figure 5 plots recognition rate of proposed PCA-LBP and PCA-CoALBP as a function of dimension of image patch subspace. As can be seen, dimension selection of subspace representation of image patch does have a effect on face recognition performance. It also indicates that face representation performance will not be improved when dimension of patch descriptor is more than 6. In fact, 6 is nearly 25 % of original dimension of image patch with size  $5 \times 5$  pixels. This observation seems to be consistent with the theorem of canonical preprocessing. In [7] Aapo Hyvärinen recommends that the number of retained principal components in image patch be chosen as 25% of original dimension in order to avoid aliasing problem. Virtually, that number of retained principal components is the dimension of image patch subspace.

**Result.** Table 1 shows the experimental result. PCA-LBP achieves 96.89% recognition rate with parameters  $\Delta r = 3$  and  $N = 6$ . And PCA-CoALBP achieves 98.95% accuracy with parameters  $\Delta r = 2$ ,  $\Delta p = 4$ ,  $\text{config} = 2$  and  $N = 4$ . It shows that our proposed method PCA-LBP and PCA-CoALBP achieved a significant improvement compared to original LBP and CoALBP. Also, it is worthwhile to note that PCA-CoALBP outperforms many state-of-art methods such as P-LBP, CELDP and PCANet-1.

**Table 1.** Experiment Result in Extended Yale Face B Database

Method	Accuracy (%)
LBP [1]	73.86
PCA-LBP	96.89
CoALBP [11]	86.70
PCA-CoALBP	<b>98.95</b>
PCANet-1 [3]	97.77
P-LBP [15]	96.13
CELDP [5]	94.55

## 4.2 Face Recognition in AR Face Database

In this experiment, we focus on attacking one sample per person problem under more variable conditions, including different occlusions, illuminations and facial expressions. To simply access the effectiveness of our methods, we only make comparison with original LBP and CoALBP.

**Database.** AR Face Database contains over 4000 images of frontal view faces with different facial expressions, illumination conditions, and occlusions(sun glasses and scarf) [10]. We use 1040 images of 40 individuals in our experiment. Figure 6 shows an example of facial images of one subject.

**Setup.** In this experiment, facial images are transformed to gray value, resized to  $126 \times 126$  pixels and divided into  $7 \times 7$  non-overlapped subregions. 40 face images (one sample per person) with frontal-lighting and neural-expressing are selected as the reference set, rest 1000 images are used as the testing set. The image patches in reference gallery is used for learning principal components in facial image patch. And 1-NN classifier based on L1 distance is used for classification.

**Result.** Table 2 shows the experiment result. PCA-LBP with parameters  $\Delta r = 3$  and  $N = 4$  achieves 96.9 % recognition rate . And proposed PCA-CoALBP achieves 95.6 % with parameters  $\Delta r = 1$ ,  $\Delta p = 4$ ,  $\text{config} = 1$  and  $N = 4$ .





**Fig. 6.** Examples in AR Face Database

Both of them outperform the original LBP and CoALBP. In addition, we observe that PCA-LBP outperforms PCA-CoALBP in this experiment. It seems related with the problem of sparse configuration in CoALBP, which makes it sensitive to noise.

**Table 2.** Experiment result in AR face database

Method	Accuracy (%)
LBP [1]	92.4
PCA-LBP	<b>96.9</b>
CoALBP [11]	91.4
PCA-CoALBP	95.6

## 5 Conclusion and Discussion

In this paper, we have proposed two local descriptors (PCA-LBP and its variant PCA-CoALBP) for face recognition. In contrast to classic LBP methods, which make intensity comparison between the central pixel and its neighborhood pixels, our proposed descriptors are obtained by comparing central image patch with its neighbors about their subspace representations. Several LBP operators based on subspace representation of image patch make it possible to incorporate more spatial information and capture macro-patterns for face recognition. Experiments in two benchmark face databases show that our proposed two methods significantly outperform classical LBP methods and achieve good results in face recognition task of one sample per person.

Moreover, our proposed method can be generically described as a hybrid framework, combining the classic local descriptor in pixel level with the learned descriptor in image patch level. This characteristic makes it possible and flexible to be transferred. (e.g. PCA-CoALBP is a transferred version of PCA-LBP). Therefore, it might also be of interest to investigate other possible combinations between various hand-craft local descriptors in pixel level and variant learned descriptors in image patch level.

## References

1. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: application to face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(12), 2037–2041 (2006)
2. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(8), 1798–1828 (2013)
3. Chan, T.H., Jia, K., Gao, S., Lu, J., Zeng, Z., Ma, Y.: PCANet: a simple deep learning baseline for image classification? *IEEE Trans. Image Process.* **24**(12), 5017–5032 (2015)
4. Fan, B., Wang, Z., Wu, F.: *Local Image Descriptor: Modern Approaches*. Springer, Heidelberg (2015). <https://doi.org/10.1007/978-3-662-49173-7>
5. Faraji, M.R., Qi, X.: Face recognition under varying illuminations using logarithmic fractal dimension-based complete eight local directional patterns. *Neurocomputing* **199**, 16–30 (2016)
6. Hintze, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. *Neural Comput.* **18**(7), 1527–1554 (2006)
7. Hyvärinen, A., Hurri, J., Hoyer, P.O.: *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision*. Springer, Heidelberg (2009). <https://doi.org/10.1007/978-1-84882-491-1>
8. Kannala, J., Rahtu, E.: BSIF: binarized statistical image features. In: *Proceedings of the 21st International Conference on Pattern Recognition (ICPR 2012)*, pp. 1363–1366, November 2012
9. Lee, K.C., Ho, J., Kriegman, D.J.: Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(5), 684–698 (2005)
10. Martinez, A.M.: The AR face database. CVC Technical Report24 (1998)
11. Nosaka, R., Ohkawa, Y., Fukui, K.: Feature extraction based on co-occurrence of adjacent local binary patterns. In: Ho, Y.-S. (ed.) *PSIVT 2011*. LNCS, vol. 7088, pp. 82–91. Springer, Heidelberg (2011). [https://doi.org/10.1007/978-3-642-25346-1\\_8](https://doi.org/10.1007/978-3-642-25346-1_8)
12. Ojala, T., Pietikainen, M., Maenpää, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 971–987 (2002). <https://doi.org/10.1109/TPAMI.2002.1017623>
13. Pietikäinen, M., Zhao, G.: Two decades of local binary patterns: a survey. *CoRR* abs/1612.06795 (2016). <http://arxiv.org/abs/1612.06795>
14. Tan, X., Chen, S., Zhou, Z.H., Zhang, F.: Face recognition from a single image per person: a survey. *Pattern Recogn.* **39**(9), 1725–1745 (2006)
15. Tan, X., Triggs, B.: Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Trans. Image Process.* **19**(6), 1635–1650 (2010)
16. Tian, Y., Fan, B., Wu, F., et al.: L2-Net: deep learning of discriminative patch descriptor in Euclidean space. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2 (2017)



# An Image-Based Representation for Graph Classification

Frédéric Rayar<sup>(✉)</sup> and Seiichi Uchida

Kyushu University, Fukuoka 819-0395, Japan  
{rayar, uchida}@human.ait.kyushu-u.ac.jp

**Abstract.** This paper proposes to study the relevance of image representations to perform graph classification. To do so, the adjacency matrix of a given graph is reordered using several matrix reordering algorithms. The resulting matrix is then converted into an image thumbnail, that is used to represent the graph. Experimentation on several chemical graph data sets and an image data set show that the proposed graph representation performs as well as the state-of-the-art methods.

**Keywords:** Graph classification · Graph representation  
Matrix reordering · Chemoinformatics

## 1 Introduction

Graphs are efficient and powerful structures to represent real-world data in several fields, such as bioinformatics [5], social networks analysis [2] or pattern recognition [30]. Formally, a graph is an ordered pair  $G = (V, E)$ , where  $V = \{v_1, \dots, v_n\}$  is a set of vertices (or nodes), and  $E \subset V \times V$  is a set of edges that represent relations between elements of  $V$ .

Graph classification [29] is an important and still challenging task, that has been widely addressed by the research community. This task falls into the supervised learning field, where one has to predict the label of an object that is represented by a graph. More formally, given a training set  $\{g_i, l_i\}$  of graphs and their labels, one has to predict the label  $l$  of an unseen graph  $g$ . Among the many studies that have been proposed to address the graph classification problem, the most used paradigms are the graph kernels [13], along with the graph edit distance [8] (GED) for error-tolerant graph matching, and more recently graph neural networks [17]. However, these paradigms face tough challenges such as the computational requirement when performing pairwise graph comparison, which is emphasised when dealing large data sets. Regarding neural networks, despite the efforts from the research community, the adaptation of convolution and pooling operations is non-trivial for non-Euclidean objects such as graphs, and still remains a challenge.

In this paper, we propose a novel image-based representation to describe graphs, and leverage this descriptor to perform fast graph classification, while

obtaining accuracies comparable with the state-of-the-art methods. The rest of the paper is organised as follows: Sect. 2 presents an overview of graph classification and graph visualisation paradigms. Section 3 details the proposed framework to obtain a graph’s image representation. The experimentation setup is given in Sect. 4 and the results that have been obtained are discussed in Sect. 5. Finally, we conclude this study in Sect. 6.

## 2 Related Works

### 2.1 Graph Classification

Many solutions can be found in the literature to perform graph classification. These methods often boil down to compare graphs between them, and the matching can be done in either:

1. *a vector space*: in this paradigm, one aims to represent a graph in a vector space to take advantage of statistical approaches. Often referred as graph embedding, a mapping  $\phi$  function projects the graph in  $\mathbb{R}^n$ :

$$\begin{aligned} \phi : G &\rightarrow \mathbb{R}^n \\ g &\mapsto \phi(g) = (f_1, \dots, f_n). \end{aligned}$$

Several approaches can be used, such as: (i) feature extraction [26] (*e.g.* number of nodes, number of edges, average degree of the nodes, number of cycles with a certain length, ...), (ii) spectral method [18] or (iii) dissimilarity representation [23] (based on distances to a set of prototype graphs).

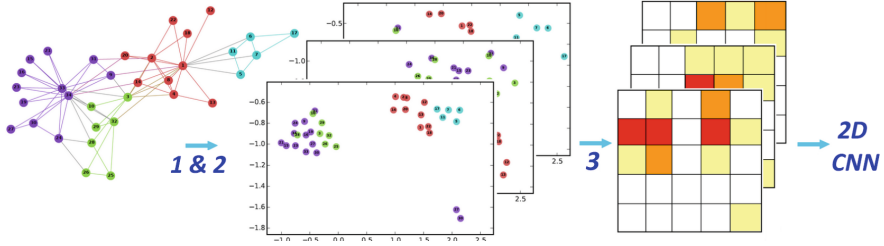
2. *the graph space*: in this paradigm, one uses graph matching methods to compare graphs in their original space. For instance, GED [8] is a well-known error-tolerant inexact graph matching algorithm. Given a set of graph edit operations (commonly insertion, deletion, substitution), the graph edit distance between two graphs  $g_1$  and  $g_2$  is given by:

$$GED(g_1, g_2) = \min_{(e_1, \dots, e_k) \in \mathcal{P}(g_1, g_2)} \sum_{i=1}^k c(e_i),$$

where  $\mathcal{P}(g_1, g_2)$  is the set of edit paths to transform  $g_1$  into  $g_2$  and  $c(e)$  is the cost of a graph edit operation  $e$ .

3. *a kernel space*: here, one leverages the kernel trick [15] to compute a similarity measure between two graphs. Kernel methods provide an implicit graph embedding and use various type of kernel, such as: random walk kernel [31], shortest-path kernel [4] or graphlet kernel [25]. One main limitation of such methods is that the extracted features are often not independent [32].

More recently, the performance of artificial neural networks has motivated their usage for graph classification. Three approaches can be considered:



**Fig. 1.** Tixier et al. framework. First, a node embedding is done along with a PCA compression (1 & 2). Then, 2D histograms are extracted and stacked to build a multi-channel image-like structure (3). Illustration from the original paper [28].

1. adapting the architecture of convolutional neural networks (CNN) to deal with graph structures (*e.g.* [20]),
2. building architecture dedicated to networks (*e.g.* [24]),
3. image-based graph representation: *i.e.* using an actual image representation along with a CNN.

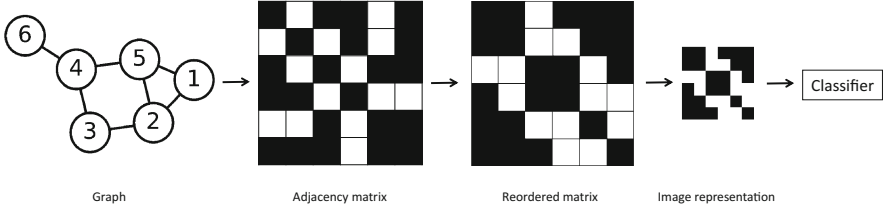
This latter approach is the first motivation of this work: computing an image representation from a graph and leverage it to use a vanilla CNN. To the best of our knowledge, only one study [28], parallel to ours and recently submitted to the arXiv repository, adopts this strategy. Indeed, in [28], Tixier et al. compute “*a multi channel image-like structure to represent a graph*”. The following steps are performed: (i) graph node embedding using node2vec [14], (ii) embedding space compression using Principal Component Analysis (PCA) and (iii) computation of fixed-size 2D histograms (that will be considered as the channels of the final image-like structure). Figure 1 illustrates their proposed framework. Even if their framework achieves classification accuracies that are comparable to baseline on several data sets, the embedding of nodes is a non-trivial step, and many parameters have to be tuned (number of channel, node2vec parameters, ...).

Hence, in this study, we propose to take advantage of existing graph visualisation techniques to build a relevant image representation for graph classification, without the need of numerous parameters.

## 2.2 Graph Visualisation

Graph drawing is a field that addresses the issue of visual depiction of graphs in two (or three) dimensional surfaces. To do so, it takes benefit of graph theory and information visualisation fields. There is two common ways to draw graphs:

- *node-link diagrams*: in such depictions, vertices of the graph are represented as disks, boxes, or textual labels. The edges are represented as segments or curves in the plane. Producing aesthetic visualisations, it is the most commonly used visualisation for graph. However, it suffers of limitations such as overlapping nodes, edge-crossing, or slow interaction for large graphs.



**Fig. 2.** Proposed framework. To represent a graph as an image, we: (i) build its adjacency matrix, (ii) apply a matrix reordering algorithm on the adjacency matrix, and (iii) convert the resulting reordered matrix into an image with predefined dimensions. This thumbnail is then given to a classifier to predict its label.

- *matrix-based visualisations*: here, the adjacency matrix of the graph is visualised. It is rarely used and most users are not familiar with this depiction, despite its “*outstanding potential*” according to [12]. Its main limitation is the fact that this visualisation is sensible to the node ordering and may produced different matrices for two graphs that have the same structure.

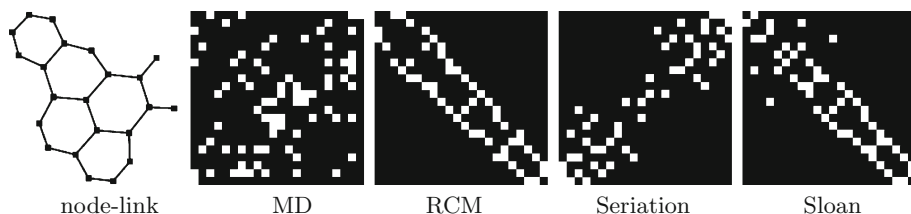
### 3 Proposed Framework

In this study, we propose to use a matrix-based visualisation of a graph and convert it to an image. This image-based representation is then be reshaped into a vector a given to classic classifier (such as k-nearest neighbour or support vector machines (SVM)) or directly feed a CNN.

Figure 2 illustrates the proposed framework. First, the adjacency matrix is extracted from the graph. We build a binary matrix  $A \in \mathfrak{M}_n$ , where  $a_{i,j} = 1$  if there is an edge between vertices  $v_i$  and  $v_j$ , 0 otherwise. Second, a matrix reordering algorithm is applied on the original adjacency matrix. An image version of the reordered matrix is built, and normalised to a predefined and fixed dimensions. A classic linear interpolation algorithm was used in our study. This final thumbnail is the proposed image-based representation of the graph.

The second step, that consists in applying a matrix reordering algorithm allows us to address the issue of the matrix-based visualisation node ordering sensibility. This will make the representation non-stochastic and also maintain spatial relevance in the obtained image. In this study, we investigate several approaches to reorder matrices, that have been selected according to two studies [3,19] on matrix reordering methods for graph visualisation. Indeed, the results of theses algorithms generally present perceivable and interpretable patterns, while heuristic implementations can be found in the literature to tackle their complexity. Namely, we investigate the following algorithms:

1. *minimum degree algorithm* [10] (MD): in numerical linear algebra, this algorithm is used to permute the rows and columns of a symmetric sparse matrix, before applying the *Cholesky decomposition*.



**Fig. 3.** Image representations of “4, 5-dimethylbenzo[a]pyrene\_sloan” molecule appearing in the PAH data set. From left to right: a node-link diagram obtained using the Fruchterman-Reingold algorithm [7] and proposed thumbnails using minimum degree, reverse Cuthill-McKee, Seriation and Sloan matrix reordering algorithms.

2. *reverse Cuthill-McKee algorithm* (RCM): the Cuthill-McKee [6] and the reverse Cuthill-McKee [11] algorithm both aim at reducing the *bandwidth* of sparse matrices.
3. *a seriation algorithm* [16] (Seriation): introduced by specialists of archaeology and palaeontology, it boils down to finding the best enumeration order of a set of objects according to a given correlation function (*e.g.* characteristic of the data, chronological order or sequential structure within the data).
4. *Sloan algorithm* [27] (Sloan): this reordering algorithm aims at reducing the *profile* and the *wavefront* of a graph. A main advantage of this algorithm is that it takes into account both global and local criteria for the reordering process.

We refer the interested readers to [3] for a more thorough survey and details on reordering algorithms. Figure 3 illustrates the different image representations obtained using the four aforementioned matrix reordering algorithms, for a given graph.

## 4 Experimental Setup

### 4.1 Data Sets

Four real-world graph data sets have been used in our experimentation:

1. GREC: this data set consists of a subset of a symbol image database. It is composed of 1100 graphs, spread among 22 classes.
2. MAO: this data set is composed of 68 molecules divided into 2 classes: molecules that inhibit the monoamine oxidase (antidepressant drugs) and molecules that do not.
3. MUTA: this data set consists in 4, 337 molecules, divided in 2 classes: mutagen and nonmutagen.
4. PAH: this data set is composed of 94 molecules, also divided in 2 classes: cancerous or not cancerous molecules.

These data sets are publicly available from the IAM Graph Database Repository [22] or the GREYC's Chemistry dataset<sup>1</sup>. The 3 first data sets are weighted and both nodes and edges are labelled. Only the PAH data set can be viewed as unweighed and not labelled, since all atoms (nodes) are carbons and all bounds (edges) are aromatics. However, for all the four data sets, we discard the weight and the nodes/edges labels. This boils down to focusing on the structure of the graphs, and generates binary adjacency matrix (1 if there is an edge, else 0), and thus binary image representation of the graphs. This choice is justified by the fact that the present study aims at evaluating the relevance of the proposed image-based representation for graph classification. In future works, greyscale and multi-channel images will be considering to handle edge weights and node/edge labels.

## 4.2 Implementation

All graphs input are in .gxl format and can be viewed using the online GXL Viewer platform<sup>2</sup>. Regarding the algorithm, we have used the C++ boost (1.58.00) graph library<sup>3</sup> implementation of the minimum degree, the reverse Cuthill-McKee and the Sloan algorithms. For the Seriation algorithm, we have used the R seriation package<sup>4</sup>.

Once the image versions of the reordered matrix are obtained, we resize them to a fixed sized of  $28 \times 28$ . This was inspired by our former goal of using CNN. Indeed, CNN performs very well on MNIST<sup>5</sup>, an isolated handwritten digits data set, that has  $28 \times 28$  images. We did not investigate the sensibility of the sole parameter of our approach at the present time.

Regarding the classifiers, we have used in these first experiments the 1-nearest neighbour (1-NN) and the 3-nearest-neighbour (3-NN) classifiers. Experiments have been done on both given train/test data sets for fair comparison with state-of-the-art results but also on the whole data set (with 10-fold cross-validation) for more generalised results.

## 5 Results and Discussion

### 5.1 Comparison with GDC 2016

During the ICPR 2016 conference, the Graph Distance Contest (GDC 2016)<sup>6</sup> has been held. Two challenges have been proposed: (1) computation of the exact or an approximate graph edit distance and (2) computation of a dissimilarity

<sup>1</sup> <https://brunl01.users.greyc.fr/CHEMISTRY/index.html>.

<sup>2</sup> <http://rfai.li.univ-tours.fr/PublicData/gxlviewer/>.

<sup>3</sup> [https://www.boost.org/doc/libs/1\\_58\\_0/libs/graph/doc/sparse\\_matrix\\_ordering.html](https://www.boost.org/doc/libs/1_58_0/libs/graph/doc/sparse_matrix_ordering.html).

<sup>4</sup> <https://CRAN.R-project.org/package=seriation>.

<sup>5</sup> <http://yann.lecun.com/exdb/mnist/>.

<sup>6</sup> <https://gdc2016.greyc.fr/>.



**Table 1.** Classification results. The recognition rate (in percentage) for the four studied matrix reordering methods on the *GREC*, *MAO* and *MUTA* data sets. Both 1-NN and 3-NN classifier have been used, on the train/test data sets of the GDR 2016 challenge 2. The results obtained by the two participants of this challenge are also presented.

	#train/test	Classifier	MD	RCM	Seriation	Sloan	Algo 1	Algo 2
GREC	484/528	1-NN	<b>91.67</b>	90.53	90.91	91.48	-	-
		3-NN	89.58	89.20	89.20	90.53	93.39	<b>99.38</b>
MAO	32/32	1-NN	81.25	<b>87.50</b>	75.00	81.25	-	-
		3-NN	<b>84.38</b>	<b>84.38</b>	68.75	71.88	68.75	75.00
MUTA	1800/2337	1-NN	58.54	<b>61.87</b>	60.63	61.70	-	-
		3-NN	57.60	64.18	59.35	61.45	<b>73.50</b>	48.55

measure for graph classification. Two participants have joined the second challenge, however, since the results of this challenge have not been published yet, we do not disclose the name of the participants, and their methods will be referred as Algo 1 and Algo 2 in the rest of the paper. The organisers of the contest kindly provided us with the results of the challenge to allow us to compare our contribution in a fair context. Only the 3-NN has been used in the challenge 2.

In order to compare the relevance of the proposed image-based representation for graph classification, we used their train/valid/test partitioning of the *GREC*, *MAO* and *MUTA* data sets (the organisers have removed 10% on the original training data sets). Since the proposed approach do not need a validation step, the classes of the test graphs are predicted using 1-NN and 3-NN classifiers on the {train;valid} subsets.

The results of this experiment are presented in Table 1. As one can see, the proposed image-based graph representations do not allow to always outperform existing methods. However, the obtained results are comparable with the one of Algo 1 and Algo 2 and for the *MAO* data set, we do indeed outperform the two participant algorithm by 10%. Furthermore, unlike our proposed representations, the participants may have used the attributes of the nodes and labels during the classification process. This supports the fact that our proposed image-based representation is a relevant graph representation for graph classification.

## 5.2 Overall Classification Accuracies

In order to generalise the results, but also to present results on the *PAH* data set, we have conducted 10-fold cross-validation experiments. Indeed, according to the organisers of the contest [1], “*PAH represented the most challenging dataset since it is composed of large unlabelled graphs*” (all nodes are carbons and all edges are aromatics).

Table 2 presents the results related to this second set of experiments. We observe the same behaviour as the previous experiments: first, the accuracies are comparable to state-of-the-art methods for the three first data sets. Regard-

**Table 2.** Classification results (2). The recognition rate (in percentage) for the four studied matrix reordering methods on the four data sets. Both 1-NN and 3-NN have been used to perform a 10-fold cross-validation technique.

	#train/test	Classifier	MD	RCM	Seriation	Sloan
GREC	990/110	1-NN	91.00	91.64	91.64	<b>92.45</b>
		3-NN	90.45	<b>91.18</b>	90.36	90.36
MAO	61/7	1-NN	79.05	<b>83.33</b>	76.19	81.90
		3-NN	<b>86.90</b>	85.24	80.95	79.52
MUTA	84/110	1-NN	62.30	<b>64.72</b>	62.35	64.26
		3-NN	59.65	<b>65.09</b>	61.59	63.15
PAH	84/110	1-NN	67.11	63.44	61.89	<b>72.56</b>
		3-NN	62.89	<b>70.00</b>	59.44	67.00

ing the PAH data set, the GREYC’s Chemistry dataset website mention the best classification accuracy achieved: 80.7% with the method presented in [9]. Second, we observe that using the 3 first nearest neighbours to classify unseen graphs do not always allow to increase the overall recognition accuracy. Finally, according to the results, even if MD and Sloan algorithms allow to have better recognition accuracies, we can not definitely conclude that a specific matrix reordering algorithm is best fit in our framework.

### 5.3 Discussion

We propose a framework where an image-based representation is leveraged to perform graph classification. The main advantage of our framework is its simplicity, that allows fast computation times while having promising accuracy results. Indeed, using greyscale or multi-channel image (without any heavy additional processes), we may considerer improving these recognition accuracies.

The major limitation of our framework, is that one does not actually compute the graph matching function, which could be a relevant asset for understanding the classification results. However, since our framework provides quickly the (dis)similarities with the training data set, one can then run a graph matching algorithm on the  $K$  first nearest neighbours in a parallel scheme, and then visualise the obtained matching with a platform such as the one proposed by [21].

## 6 Conclusion

The main contribution of this study is to show the feasibility of using a simple yet relevant image-based representation for graph classification. Our approach allows to obtain recognition accuracies that are comparable or better than the state-of-the-art methods, while avoiding the complexity of these methods.

These promising first results allow to consider several future works: (i) the usage of greyscale and multi-channel images, to take into account edge weights

and nodes/edges labels (the latter being more challenging), (ii) the usage of a combination of images to represent a graph, or boosting technique, (iii) the usage of another classifier such as SVM or CNN, that may allow to increase the recognition accuracies. Finally, it could be interesting to apply our framework on the data sets used by Tixier et al., to compare our approaches.

**Acknowledgement.** The authors would like to give credits to the organisers of the Graph Distance Contest, who provided the challenge data sets and the results of the second challenge. This research was partially supported by MEXT-Japan (Grant No. 17H06100).

## References

1. Abu-Aisheh, Z., et al.: Graph edit distance contest. *Pattern Recogn. Lett.* **100**(C), 96–103 (2017)
2. Barnes, J., Harary, F.: Graph theory in network analysis. *Soc. Netw.* **5**(2), 235–244 (1983)
3. Behrisch, M., Bach, B., Riche, N.H., Schreck, T., Fekete, J.: Matrix reordering methods for table and network visualization. *Comput. Graph. Forum* **35**(3), 693–716 (2016)
4. Borgwardt, K.M., Kriegel, H.P.: Shortest-path kernels on graphs. In: *Proceedings of the Fifth IEEE International Conference on Data Mining*, pp. 74–81. IEEE Computer Society (2005)
5. Chacko, E., Ranganathan, S.: *Graphs in Bioinformatics*, pp. 191–219. Wiley, Hoboken (2010). Chap. 10
6. Cuthill, E., McKee, J.: Reducing the bandwidth of sparse symmetric matrices. In: *Proceedings of the 1969 24th National Conference*, pp. 157–172. ACM (1969)
7. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. *Softw. Pract. Exper.* **21**(11), 1129–1164 (1991)
8. Gao, X., Xiao, B., Tao, D., Li, X.: A survey of graph edit distance. *Pattern Anal. Appl.* **13**(1), 113–129 (2010)
9. Gaüzère, B., Brun, L., Villemin, D.: Graph kernel encoding substituents’ relative positioning. In: *International Conference on Pattern Recognition* (2014)
10. George, A., Liu, J.W.: The evolution of the minimum degree ordering algorithm. *SIAM Rev.* **31**(1), 1–19 (1989)
11. George, J.A.: *Computer implementation of the finite element method*. Ph.D. thesis. Stanford, CA, USA (1971)
12. Ghoniem, M., Fekete, J.D., Castagliola, P.: On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. *Inf. Vis.* **4**(2), 114–135 (2005)
13. Ghosh, S., Das, N., Gonçalves, T., Quaresma, P., Kundu, M.: The journey of graph kernels through two decades. *Comput. Sci. Rev.* **27**, 88–111 (2018)
14. Grover, A., Leskovec, J.: Node2vec: scalable feature learning for networks. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864. ACM (2016)
15. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel methods in machine learning. *Anna. Stat.* **36**(3), 1171–1220 (2008)
16. Ihm, P.: A contribution to the history of seriation in archaeology. In: Weihs, C., Gaul, W. (eds.) *Classification - the Ubiquitous Challenge*, pp. 307–316. Springer, Heidelberg (2005). <https://doi.org/10.1007/3-540-28084-7-34>

17. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint [arXiv:1609.02907](https://arxiv.org/abs/1609.02907) (2016)
18. Luo, B., Wilson, R.C., Hancock, E.R.: Spectral embedding of graphs. *Pattern Recogn.* **36**(10), 2213–2230 (2003)
19. Mueller, C., Martin, B., Lumsdaine, A.: A comparison of vertex ordering algorithms for large graph visualization. In: 2007 6th International Asia-Pacific Symposium on Visualization, pp. 141–148 (2007)
20. Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. CoRR abs/1605.05273 (2016). <http://arxiv.org/abs/1605.05273>
21. Rayar, F., Abu-Aisheh, Z.: Photo(Graph) Gallery: An “exhibition ” of graph classification. In: International Conference on Information Visualisation (2017)
22. Riesen, K., Bunke, H.: IAM graph database repository for graph based pattern recognition and machine learning. *Pattern Recogn. Lett.* **5342**, 287–297 (2008)
23. Riesen, K., Bunke, H.: Graph Classification and Clustering Based on Vector Space Embedding. World Scientific Publishing Co., Inc., Singapore (2010)
24. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. *IEEE Trans. Neural Netw.* **20**(1), 61–80 (2009)
25. Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., Borgwardt, K.: Efficient graphlet kernels for large graph comparison. In: International Conference on Artificial Intelligence and Statistics, vol. 5, pp. 488–495. PMLR (2009)
26. Sidere, N., Heroux, P., Ramel, J.Y.: A vectorial representation for the indexation of structural informations. In: da Vitoria Lobo, N., et al. (eds.) *Structural, Syntactic, and Statistical Pattern Recognition*, pp. 45–54. Springer, Heidelberg (2008). [https://doi.org/10.1007/978-3-540-89689-0\\_9](https://doi.org/10.1007/978-3-540-89689-0_9)
27. Sloan, S.W.: An algorithm for profile and wavefront reduction of sparse matrices. *Int. J. Numer. Methods Eng.* **23**(2), 239–251 (1986)
28. Tixier, A.J., Nikolentzos, G., Meladianos, P., Vazirgiannis, M.: Classifying graphs as images with convolutional neural networks. CoRR abs/1708.02218 (2017). <http://arxiv.org/abs/1708.02218>
29. Tsuda, K., Saigo, H.: Graph classification. In: Aggarwal, C., Wang, H. (eds.) *Managing and Mining Graph Data*, pp. 337–363. Springer, Heidelberg (2010)
30. Vento, M.: A long trip in the charming world of graphs for pattern recognition. *Pattern Recogn.* **48**(2), 291–301 (2015)
31. Vishwanathan, S.V.N., Borgwardt, K.M., Schraudolph, N.N.: Fast computation of graph kernels. In: *Proceedings of the 19th International Conference on Neural Information Processing Systems*, pp. 1449–1456. MIT Press (2006)
32. Yanardag, P., Vishwanathan, S.V.N.: Deep graph kernels. In: *KDD* (2015)



# Visual Tracking via Patch-Based Absorbing Markov Chain

Ziwei Xiong, Nan Zhao, Chenglong Li<sup>(✉)</sup>, and Jin Tang

School of Computer Science and Technology, Anhui University, Hefei, China  
xzw992@gmail.com, zhn1528@gmail.com, lc11314@foxmail.com,  
jtang99029@foxmail.com

**Abstract.** Bounding box description of target object usually includes background clutter, which easily degrades tracking performance. To handle this problem, we propose a general approach to learn robust object representation for visual tracking. It relies a novel patch-based absorbing Markov chain (AMC) algorithm. First, we represent object bounding box with a graph whose nodes are image patches, and introduce a weight for each patch that describes its reliability belonging to foreground object to mitigate background clutter. Second, we propose a simple yet effective AMC-based method to optimize reliable foreground patch seeds as their qualities are very important for patch weight computation. Third, based on the optimized seeds, we also utilize AMC to compute patch weights. Finally, the patch weights are incorporated into object feature description and tracking is carried out by adopting structured support vector machine algorithm. Experiments on the benchmark dataset demonstrate the effectiveness of our proposed approach.

**Keywords:** Visual tracking · Absorbing Markov chain  
Weighted patch representation · Seed optimization

## 1 Introduction

Visual tracking is a fundamental and active research topic in computer vision due to its various applications, such as security and surveillance, human computer interaction and self-driving system. Although many tracking algorithms have made great progress recently, it still remains many challenges in practical, including complex appearance, pose variations, partial occlusion, illumination change and background clutter.

Many efforts have been devoted to weaken the effects of undesirable background information. Some methods [3, 6, 7] simply update the object classifiers by considering the distances of samples in accordance with the bounding box center, e.g., the samples far away from the center assigning smaller weights because a farther distance means a higher possibility of being background noise. Some [13–15] develop dynamic graph to learn robust patch weights. Recently, Kim et al. [11] proposed a novel descriptor named spatially ordered and weighted

patch (SOWP), which can better describe target objects and suppress background information. The method utilizes similarities between initialized patch seeds with other image patches to represent patch weights via random walk algorithm [19]. They indeed achieve much better performance than other trackers. However, the random walk algorithm adopted in this method still has two issues as the follows: (1) it is an iterative algorithm, and (2) its performance relies on initial seeds, which are usually contagious due to inaccurate tracking results and deformation or occlusion of target objects.

To handle these issues, we propose a novel patch-based absorbing Markov chain (AMC) algorithm [9] to compute robust patch weights for visual tracking. First, we represent object bounding box with a graph whose nodes are image patches as they are robust to object deformation and partial occlusion. To mitigate background noise of patches within the bounding box, we assign a weight for each patch which describes its reliability belonging to foreground object. Second, we propose a simple yet effective AMC-based method to optimize reliable foreground patch seeds as their qualities are very important for patch weight computation. In particular, we design a criterion using the peak-to-sidelobe ratio (PSR) [17] to measure the quality of foreground patches, and then select most reliable ones as seeds for patch weight computation. Third, we utilize AMC once again to compute patch weights with the optimized seeds as inputs, and the patch weights are finally incorporated into object feature description and tracking is carried out by adopting structured support vector machine algorithm [6]. The pipeline of our approach is shown in Fig. 1.

Our approach has following advantages. First, it is able to mitigate noises of foreground patch seeds based on the AMC algorithm and PSR criterion. Second, it is efficient due to closed-form solution of AMC. Third, it achieves superior performance against SOWP and other trackers on a large-scale benchmark dataset.

## 2 Related Work

### 2.1 Visual Tracking

Here we only discuss the most related visual tracking works with ours. And comprehensive review can be found in [12, 21].

To suppress background noise, some methods [5, 22] integrate segmentation results into tracking to alleviate the effects of background. These methods, however, are sensitive to segmentation results. Some [16, 23] construct a graph for absorbing Markov chain (AMC) using superpixels in two consecutive frames or between the first frame and the current frame to estimate and propagate target segmentations in a spatio-temporal domain. Also, one representative approach is to assign weights to different pixels in the bounding box, such that [3, 7] assume pixels far away from the bounding box center should be less important, and thus assign smaller weights to boundary pixels via the kernel-based method during the histogram construction. However, these methods may be failed when a target object has a complicated shapes or is occluded. Kim et al. [11] compute patch weights within bounding box through a random walk with restart algorithm which has a high computation burden. Moreover, they simply define all

the inner patches as foreground seeds like the initial patch seeds shown in Fig. 1. It is obvious that the SOWP descriptor inevitably has some improper initial foreground seeds in this way, especially when the target object is occluded.

## 2.2 Absorbing Markov Chain

Our approach relies on absorbing Markov chain (AMC), so we describe it in detail. AMC includes two kinds of nodes, absorbing nodes and transient nodes representing absorbing states and non-absorbing states respectively. The transient nodes which have similar appearance and small spatial distance to absorbing nodes can be absorbed faster. Therefore, the absorbed time can be regarded as our patch weights because it represents the similarity between a pair of nodes.

Given  $n$  nodes  $S = \{s_1, s_2, \dots, s_n\}$  including  $r$  absorbing nodes and  $t$  transient nodes, the  $n \times n$  transition matrix  $\mathbf{P}$ , where  $p_{ij}$  is the probability of moving from node  $s_i$  to node  $s_j$ , have the following canonical form:

$$\mathbf{P} \rightarrow \begin{pmatrix} \mathbf{Q} & \mathbf{R} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}, \quad (1)$$

where the first  $t$  nodes are transient and the last  $r$  nodes are absorbing.  $\mathbf{Q} \in [0, 1]^{t \times t}$  and  $\mathbf{R} \in [0, 1]^{t \times r}$  denotes the transition probabilities between any pair of transient nodes, and transient nodes with any absorbing node respectively.  $\mathbf{0}$  is zero matrix and  $\mathbf{I}$  is identity matrix. For an absorbing chain, we can derive its fundamental matrix  $\mathbf{N} = \sum_{k=0}^{\infty} \mathbf{Q}^k = (\mathbf{I} - \mathbf{Q})^{-1}$ , which is the expected number of times that spends from the transient node  $s_i$  to the transient node  $s_j$ , and the sum  $\sum_j n_{ij}$  reveals the expected number of times before absorption. Thus, we can compute the absorbed time  $\mathbf{z}$  for each transient node by

$$\mathbf{z} = \mathbf{N} \times \mathbf{c}, \quad (2)$$

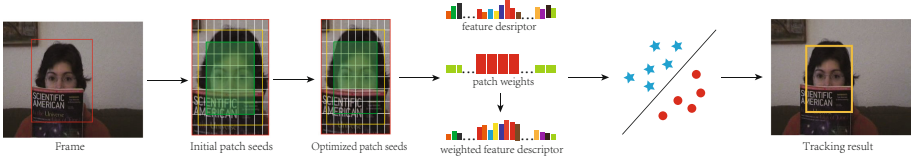
where  $\mathbf{c}$  is a  $t$  dimensional column vector all of whose elements are 1. Notice that a small  $z(i)$  means a high similarity between the  $i$ -th transient node and absorbing nodes.

## 3 Proposed Methodology

The proposed algorithm utilizes absorbing Markov chain (AMC) to reduce the impacts of background information in object representation. In this section, we describe how to use patch-based AMC to gain the patch weights. Also, we introduce our AMC-based method for foreground seed optimization in order to remove some improper foreground seeds.

### 3.1 Overview of Our Approach

Given object bounding box of an unknown target in the first frame, we first represent it with a graph which takes image patches as nodes. The graph is described



**Fig. 1.** Pipeline of our method. Input frame with patch partition, where the expanded, original and shrunk bounding boxes are indicated by red, yellow and green colors. The foreground seeds are highlighted by green color. (Color figure online)

with features constructed by a combination of Hog and RGB color histogram and used for the absorbing Markov chain (AMC). Then we use a AMC-based method to remove some improper foreground seeds because foreground seeds sometimes have a large area of background region when the target object has a complex appearance or is occluded. After that, we use AMC once again with the optimized seeds to calculate patch weights and combine these weights with corresponding patch features to construct a robust object descriptor. Finally, the descriptor can be incorporated into the Structured SVM [6] to conduct our tracking. The pipeline of our method is shown in Fig. 1.

### 3.2 Object Feature Learning with Patch-Based AMC

**Graph Representation.** We first decompose the bounding box into  $n$  non-overlapping patches and characterize each patch with low-level features. Then the spatially ordered patch feature descriptor for the bounding box is given by:  $\Phi(\mathbf{x}_t, y) = [\mathbf{f}_1^T, \dots, \mathbf{f}_n^T]^T$ , which represents the contents in a bounding box  $y$  in the  $t$ -th frame  $\mathbf{x}_t$ , and  $\mathbf{f}_i$  is the feature vector of the  $i$ -th patch.

We construct a graph  $G(V, E)$  with these patches as nodes  $V$  and the links between patches as edges  $E$ . Each node is connected with the neighboring nodes and nodes that share common boundaries with them. Then we can effectively capture local smoothness cues as neighboring patches tend to share similar appearance, and explore more intrinsic relationship among patches as the same semantic region has likely similar appearance and high compactness. The weight  $w_{ij}$  of the edge  $e_{ij}$  between adjacent nodes  $i$  and  $j$  is defined as

$$w_{ij} = \exp(-\gamma \|\mathbf{f}_i - \mathbf{f}_j\|^2) \quad (3)$$

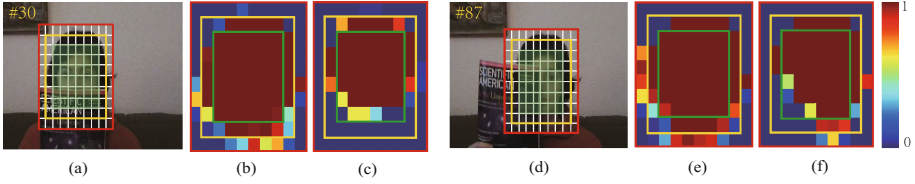
For AMC, we first renumber the nodes so that the first  $t$  nodes are transient nodes and the last  $r$  nodes are absorbing nodes. Then, the affinity matrix  $\mathbf{A}$  is defined as

$$a_{ij} = \begin{cases} w_{ij} & j \in \mathbf{N}(i), 1 \leq i \leq t \\ 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

where  $\mathbf{N}(i)$  denotes the nodes connected to node  $i$ . Therefore, we can obtain the transition matrix  $\mathbf{P}$  on the sparsely connected graph which is given as

$$\mathbf{P} = \mathbf{D}^{-1} \times \mathbf{A}, \quad (5)$$





**Fig. 2.** Illustration of effectiveness of optimized seeds for patch weight calculation. (a) and (d) Input frame with patch partition, where the expanded, original and shrunk bounding boxes are indicated by red, yellow and green colors. The patch seeds are highlighted by green color. (b) and (e) Patch weight calculation via initial seeds. (c) and (f) Patch weight calculation via the proposed optimized seeds. The results show that our method is able to handle occlusion effectively. (Color figure online)

where  $\mathbf{D} = \text{diag}(\sum_j a_{ij})$  is the degree matrix of each node that records the sum of the weights, and  $\mathbf{P}$  is actually the raw normalized  $\mathbf{A}$ . In this way, we get a patch-based AMC that can achieve a graph representation. In the next section, we will discuss our AMC-based method for foreground seed optimization.

**Foreground Seed Optimization.** Given the original bounding box, we expand and shrink it respectively as shown in Fig. 2. Then inner patches which are located inside the shrunk region are taken as initial foreground seeds. To remove some improper foreground seeds such that the seeds contain a large area of background, specifically, we select only one inner patch as absorbing node one time, and all the other patches as transient nodes. The corresponding absorbed time can be obtained by the following steps: (a) Get the affinity matrix  $\mathbf{A}$  by Eq. (4); (b) Calculate the transition matrix  $\mathbf{P}$  by Eq. (5); (c) Extract the matrix  $\mathbf{Q}$  by Eq. (1); (d) Compute the fundamental matrix  $\mathbf{N}$ ; (e) Compute the absorbed time  $\mathbf{z}$  by Eq. (2) and normalize it to the range between 0 and 1.

Then we adopt PSR based on AMC as a confidence metric to remove some improper seeds, which is widely used in signal processing to measure the signal peak strength in a response map. Inspired by [1, 17], we generalize the PSR as a confidence function for the candidate seed as:

$$PSR_{s_i} = \frac{\max \rho_{s_i} - \mu_{\Omega, s_i}}{\sigma_{\Omega, s_i}} \quad (6)$$

where  $s_i$  is the  $i$ -th candidate seed as absorbing node in a Markov chain and  $\rho_{s_i}$  is its probability map (normalized absorbed time).  $\Omega$  is the sidelobe area around the peak which is 36% of the probability map area in this paper.  $\mu_{\Omega, s_i}$  and  $\sigma_{\Omega, s_i}$  are the mean value and standard deviation of  $\rho_{s_i}$  except area  $\Omega$  respectively. It can be easily seen that the function  $PSR_{s_i}$  becomes large when the probability peak is strong. Therefore,  $PSR_{s_i}$  can be treated as the confidence function to measure whether the candidate seed can be a seed properly. When  $PSR_{s_i} < \text{threshold}$ , we make the  $i$ -th improper absorbing node to be a transient node, otherwise keep it unchanged. In this way, we can obtain the optimized foreground

seeds. As shown in Fig. 2, the distribution of patch weights with foreground seed optimization in Fig. 2 (c) and (f) is more accurate than the method without foreground seed optimization in Fig. 2 (b) and (e).

**Patch Weight Calculation.** After we obtain the optimized foreground seeds, and take outer patches, which are located inside the expanded region but outside the original region as background seeds, we can calculate the final patch weights. At first, the optimized foreground seeds are taken as absorbing nodes and other patches are taken as transient nodes. Then we can calculate the foreground normalized absorbed time through steps (a) – (e) mentioned above and get a normalized absorbed time vector  $\bar{\mathbf{z}}^F = [\bar{z}^F(1), \bar{z}^F(2), \dots, \bar{z}^F(n)]$ . Then in turn we take background seeds as absorbing nodes and others as transient nodes and have the background absorbed time  $\bar{\mathbf{z}}^B = [\bar{z}^B(1), \bar{z}^B(2), \dots, \bar{z}^B(n)]$ . Thus, for the  $i$ -th patch at the  $t$ -th frame, we compute the final patch weight  $z_t(i)$  by combining the foreground absorbed time with background absorbed time:

$$z_t(i) = \frac{1}{1 + \exp(-\beta(\bar{z}_t^F(i) - \bar{z}_t^B(i)))}. \quad (7)$$

where  $\beta$  controls the steepness of the logistic function. Thus, we incorporate the patch weights with the feature descriptor, and consequently obtain our robust weighted feature descriptor  $\Phi(\mathbf{x}_t, y) = [z_t(1)\mathbf{f}_1^T, \dots, z_t(n)\mathbf{f}_n^T]^T$ . In Fig. 2 we can find that the patches, which are assigned relatively large weights, reveal the shape of the target object effectively.

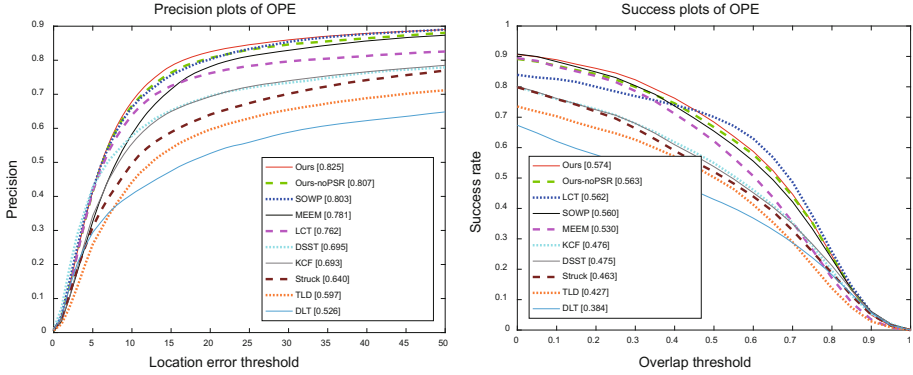
### 3.3 Structured SVM Tracking

Given the bounding box of the target object in the previous frame  $t - 1$ , we first set a searching window in the current frame  $t$ . For  $i$ -th candidate bounding box within the search window, we obtain its weighted feature descriptor by the proposed patch-based AMC algorithm and incorporate it into the conventional tracking-by-detection algorithm, Struck [6]. Note that in addition to Struck, there are other tracking-by-detection algorithms, such as [2, 25], can also be combined with our descriptor for tracking. We also adopt the schemes of scale estimation [18] and model update [11] to handle scale variations and avoid drastic appearance changes.

## 4 Experimental Results

### 4.1 Implementation

The proposed method is implemented in C++ on an Intel I7-6770K 4 GHz CPU with 32 GB RAM. We set 0.3 as the confidence score threshold, and the parameters are empirically set as  $\gamma = 5.0$  in Eq. (3),  $\beta = 30$  in Eq. (7) and  $threshold = 3.0$  for foreground optimization. The side length of a searching window is fixed to  $2\sqrt{WH}$ , where  $W$  and  $H$  are the width and height of the scaled bounding box respectively.



**Fig. 3.** Evaluation results on the OTB100 benchmark. The representative score of PR/SR is presented in the legend.

### 4.2 OTB100 Benchmark Dataset

We evaluate the proposed tracking method on the OTB100 benchmark dataset [21] which contains 100 videos with ground-truth object locations and different attributes for performance analysis. We use distance precision rate (PR) and overlap success rate (SR) with the threshold of 20 pixels for quantitative performance.

### 4.3 Evaluation on OTB100

We compare the performances of our proposed algorithm with other conventional trackers whose results were reported in [11, 21] including MEEM [24], LCT [18], DSST [4], KCF [8], Struck [6], TLD [10], DLT [20] and SOWP [11]. The precision and success rate are presented in Fig. 3. Also, the results of attribute-based evaluation are showed in Table 1.

**Overall Comparison:** As shown in Fig. 3, our proposed method shows a superior performance against SOWP and outperforms other conventional methods significantly. In particular, our tracker outperforms SOWP with 2.2%/1.4% in precision and success rates respectively. That means our method has a more robust descriptor compared with SOWP and can better reduce the influence of background information. In summary, the precision and success plots demonstrate that our method performs well against these conventional methods.

**Attribute-Based Comparison:** We compare the precision and success scores of our algorithm with the conventional trackers over 11 challenging factors in Table 1. We can find that the proposed method performs favorably against conventional trackers and always yields the top three scores in both precision and success metrics. Specifically, most of our top scores are over 1% higher than second place. There are also some issues that we can easily notice as follows: The SOWP method does not perform well during fast motion and motion blur

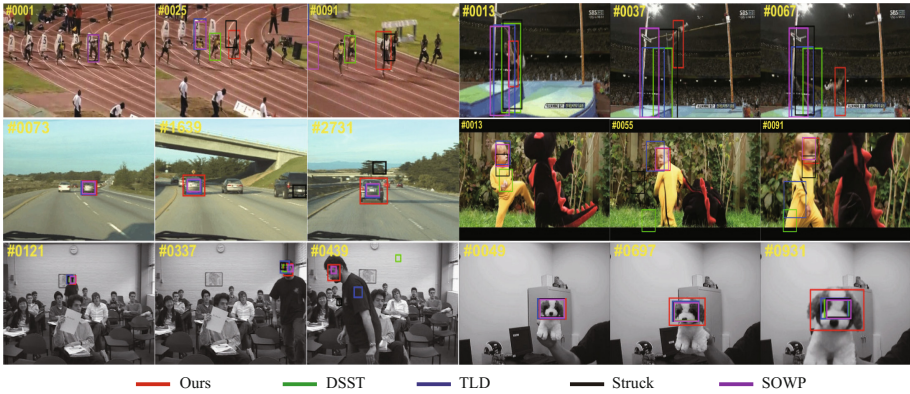
**Table 1.** Precision rate and success rate based on differ attributes of OTB100 benchmark [21] with recent 8 trackers. The attributes include scale variation (SV), fast motion (FM), background clutter (BC), motion blur (MB), deformation (DF), illumination variation (IV), in-plane rotation (IPR), low resolution (LR), occlusion (OC), out-of-plane rotation (OPR), out of view (OV). The best, second and third results are in red, green and blue colors, respectively.

	MEEM	LCT	DSST	KCF	Struck	DLT	SOWP	Ours
SV	73.6/47.0	68.1/48.8	66.2/40.9	63.6/39.6	60.0/40.4	53.5/39.1	74.6/47.5	77.2/50.8
FM	75.2/54.2	68.1/53.4	58.4/44.2	62.5/46.3	62.6/47.0	39.1/31.8	72.3/55.6	78.9/57.7
BC	74.6/51.9	73.4/55.0	70.2/47.7	71.8/50.0	56.6/43.8	51.5/37.2	77.5/57.0	78.5/58.3
MB	73.1/55.6	66.9/53.3	61.1/46.7	60.6/46.3	59.4/46.8	38.7/32.0	70.2/56.7	77.3/58.2
DF	75.4/48.9	68.9/49.9	56.8/41.2	61.7/43.6	52.7/38.3	45.1/29.5	74.1/52.7	83.7/56.3
IV	72.8/51.5	73.2/55.7	70.8/48.5	69.3/47.1	54.5/42.2	51.5/40.1	76.6/55.4	77.0/54.3
IPR	79.4/52.9	78.2/55.7	72.4/48.5	69.7/46.7	63.7/45.3	47.1/34.8	82.8/56.7	80.7/55.3
LR	80.8/38.2	69.9/39.9	70.8/31.4	67.1/29.0	67.4/31.3	75.1/46.5	90.3/42.3	79.9/40.7
OC	74.1/50.4	68.2/50.7	61.5/42.6	62.5/44.1	53.7/39.4	45.4/33.5	75.4/52.8	76.2/53.1
OPR	79.4/52.5	74.6/53.8	67.0/44.8	67.0/45.0	59.3/42.4	50.9/37.1	78.7/54.7	79.8/54.6
OV	68.5/48.8	59.2/45.2	48.7/37.4	51.2/40.1	50.3/38.4	55.8/38.4	63.3/49.7	73.0/53.1
ALL	78.1/53.0	76.2/56.2	69.5/47.5	69.3/47.6	64.0/46.3	52.6/38.4	80.3/56.0	82.5/57.4

or when the object is out of view. The MEEM method can not handle partial occlusion well. The LCT and DSST methods do not perform well when the object is out of view. And the DSST method drifts when fast motion happens or the object has a complex deformation. The KCF and Struck methods have a bad tracking result when target objects suffer from heavy occlusion and fast motion. But overall it is obvious that our proposed algorithm can well handle different challenging factors. And that is because we give the classifier a more robust descriptor of target objects. We can see our tracking examples in Fig. 4.

#### 4.4 Ablation Study

As shown in Fig. 3, our method with foreground seed optimization via PSR has a higher precision and success rate curves than the method without it. The reason is that the initial foreground seeds may have a large area of background noise due to complex appearance or partial occlusion. It indicates that our method can suppress background noise effectively. And it confirms our scheme of using optimized foreground seeds can get a more robust patch weights and construct a more reliable descriptor. Also, our method is 6.63-fps, a little lower than 8.26-fps in SOWP because although absorbing Markov chain has a closed-form solution, our AMC-based method for foreground seed optimization has to determine the reliability of each initial foreground seed.



**Fig. 4.** The tracking results of the proposed method with other conventional trackers on OTB100 benchmark.

## 5 Conclusion

In this paper, we propose an effective approach to learn robust object representation for visual tracking via a patch-based absorbing Markov chain algorithm with foreground seed optimization. Note that the optimized foreground seeds make great contributions for a more robust patch weights calculation. Experiments on benchmark dataset demonstrate the effectiveness and robustness of the proposed algorithm. In future work, we will improve the efficiency of our approach and introduce more robust features.

**Acknowledgment.** This work was jointly supported by National Natural Science Foundation of China (61702002, 61472002), Natural Science Foundation of Anhui Province (1808085QF187), Natural Science Foundation of Anhui Higher Education Institution of China (KJ2017A017) and Co-Innovation Center for Information Supply & Assurance Technology of Anhui University.

## References

1. Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M.: Visual object tracking using adaptive correlation filters. In: IEEE Conference CVPR, pp. 2544–2550 (2010)
2. Chen, D., Yuan, Z., Hua, G., Wu, Y., Zheng, N.: Description-discrimination collaborative tracking. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 345–360. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-10590-1\\_23](https://doi.org/10.1007/978-3-319-10590-1_23)
3. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. TPAMI **25**, 564–577 (2003)
4. Danelljan, M., Hager, G., Khan, F., Felsberg, M.: Accurate scale estimation for robust visual tracking. In: Proceedings BMVC (2014)
5. Duffner, S., Garcia, C.: Pixeltrack: a fast adaptive algorithm for tracking non-rigid objects. In: Proceedings IEEE Conference ICCV (2013)

6. Hare, S., Saffari, A., Torr, P.H.S.: Struck: structured output tracking with kernels. In: Proceedings IEEE Conference ICCV (2011)
7. He, S., Yang, Q., Lau, R., Wang, J., Yang, M.H.: Visual tracking via locality sensitive histograms. In: Proceedings IEEE Conference CVPR (2013)
8. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. *TPAMI* **37**, 583–596 (2015)
9. Jiang, B., Zhang, L., Lu, H., Yang, C., Yang, M.H.: Saliency detection via absorbing markov chain. In: Proceedings IEEE Conference ICCV (2013)
10. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. *TPAMI* **34**(7), 1409–1422 (2012)
11. Kim, H.U., Lee, D.Y., Sim, J.Y., Kim, C.S.: SOWP: spatially ordered and weighted patch descriptor for visual tracking. In: Proceedings IEEE Conference ICCV (2015)
12. Li, C., Liang, X., Lu, Y., Zhao, N., Tang, J.: RGB-T object tracking: benchmark and baseline. [arXiv:1805.08982](https://arxiv.org/abs/1805.08982) (2018)
13. Li, C., Lin, L., Zuo, W., Tang, J.: Learning patch-based dynamic graph for visual tracking. In: Proceedings AAAI (2017)
14. Li, C., Lin, L., Zuo, W., Tang, J., Yang, M.H.: Visual tracking via dynamic graph learning. [arXiv:1710.01444](https://arxiv.org/abs/1710.01444) (2018)
15. Li, C., Wu, X., Bao, Z., Tang, J.: ReGLe: spatially regularized graph learning for visual tracking. In: MM Proceedings ACM (2017)
16. Li, X., Han, Z., Wang, L., Lu, H.: Visual tracking via random walks on graph model. *IEEE Trans. Cybern.* **46**(9), 2144–2155 (2016)
17. Liu, T., Wang, G., Yang, Q.: Real-time part-based visual tracking via adaptive correlation filters. In: IEEE Conference CVPR (2015)
18. Ma, C., Yang, X., Zhang, C., Yang, M.H.: Long-term correlation tracking. In: Proceedings IEEE Conference CVPR (2015)
19. Tong, H., Faloutsos, C., Pan, J.Y.: Random walk with restart: fast solutions and applications. *KAIS* **14**(3), 327–346 (2008)
20. Wang, N., Yeung, D.Y.: Learning a deep compact image representation for visual tracking. In: NIPS, pp. 809–817 (2013)
21. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. *TPAMI* **37**, 1834–1848 (2015)
22. Yang, F., Lu, H., Yang, M.H.: Robust superpixel tracking. *IEEE Trans. Image Process.* **23**(4), 1639–1651 (2014)
23. Yeo, D., Son, J., Han, B., Han, J.H.: Superpixel-based tracking-by-segmentation using markov chains. In: CVPR, pp. 511–520 (2017)
24. Zhang, J., Ma, S., Sclaroff, S.: MEEM: robust tracking via multiple experts using entropy minimization. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8694, pp. 188–203. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-10599-4\\_13](https://doi.org/10.1007/978-3-319-10599-4_13)
25. Zhang, K., Zhang, L., hsuan Yang, M.: Real-time compressive tracking. In: Proceedings ECCV (2012)



# Gradient Descent for Gaussian Processes Variance Reduction

Lorenzo Bottarelli<sup>1</sup>(✉) and Marco Loog<sup>2</sup>

<sup>1</sup> Department of Computer Science, University of Verona, Verona, Italy  
lorenzo.bottarelli@univr.it

<sup>2</sup> Pattern Recognition Laboratory, Delft University of Technology,  
Delft, The Netherlands  
m.loog@tudelft.nl

**Abstract.** A key issue in Gaussian Process modeling is to decide on the locations where measurements are going to be taken. A good set of observations will provide a better model. Current state of the art selects such a set so as to minimize the posterior variance of the Gaussian Process by exploiting submodularity. We propose a Gradient Descent procedure to iteratively improve an initial set of observations so as to minimize the posterior variance directly. The performance of the technique is analyzed under different conditions by varying the number of measurement points, the dimensionality of the domain and the hyperparameters of the Gaussian Process. Results show the applicability of the technique and the clear improvements that can be obtain under different settings.

## 1 Introduction

In many analyses we are dealing with spatial phenomena modeled using Gaussian Processes (GPs, [11]). When tackling the analysis of such spatial phenomena in a data-driven manner, a key issue is to decide on the locations where measurements are going to be taken. The better the choice of locations, the better the GP will approximate the true underlying functional relationship or the fewer measurements we need to get a model to a prespecified level of performance.

One example is environmental monitoring, where it is necessary to choose a set of locations in space in which to measure the specific phenomenon of interest. Such environmental analysis processes, required to characterize and monitor the quality of the environment, typically includes two phases: (i) the collection of the information and (ii) the generation of a model to effectively predict the spatial phenomena of interest. The measurements through the use of mobile sensors [1, 2, 8] or the displacement of fixed sensors [3, 5, 7] is, however, usually costly and one would want to select observations that are especially informative with respect to some objective function.

Recent research in this context has exactly aimed at selecting such a set of measurement locations so as to minimize the posterior variance of the GP [6]. This selection of measurement locations is basically performed through the use of



greedy procedures. In particular submodularity, which is an intuitive diminishing returns property, is exploited [4, 5, 10].

Although submodular objective functions allows for a greedy optimization with bound guarantees [9], the solution that these techniques offer can deviate considerably from the optimum and there is definitely room for improvement. This is the main goal of this work: we propose a direct Gradient Descent (GD) procedure to minimize the posterior variance of the GP and present a study of its performance. We basically use a GD algorithm to adapt the sensing locations starting from a set of initial positions that can be given from any other algorithm.

The core contributions of our paper are GD approach to minimize the posterior variance of a GP and an extensive empirical evaluation of the procedure under different conditions by varying: (i) the hyperparameters of the GP; (ii) the dimensionality of the dataset; (iii) the number of points to adapt; (iv) the method of initialization of the points. Moreover, we present the results and discuss the applicability and the improvements that our technique offers. In particular, we show how submodular greedy solutions can be further improved.

The paper is organized as follows: Sect. 2 provides the required background and the problem definition. Section 3 presents our algorithm and describes its implementation. Section 4 provides the detailed description of the experimental settings and Sect. 5 presents the results. Section 6 provides a discussion and conclusions.

## 2 Background

### 2.1 Gaussian Processes

GPs are a widely used tool in machine learning [11]. A GP provides a statistical distribution together with a way to model an unknown function  $f$ .

A GP is completely defined by its mean and a kernel function (also called covariance function)  $k(x, x')$  which encodes the smoothness properties of the modeled function  $f$ . We consider GPs that are estimated based on a set  $K$  of noisy measurements  $Y = \{y_1, y_2, \dots, y_K\}$  taken at locations  $\{x_1, x_2, \dots, x_K\}$ . We assume that  $y_i = f(x_i) + e_i$  where  $e_i \sim \mathcal{N}(0, \sigma_n^2)$ , i.e., zero mean Gaussian noise. The posterior over  $f$  is then still a GP and its mean and variance can be computed as follows [11]:

$$\mu(x) = \mathbf{k}(x)^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} Y \quad (1)$$

$$\sigma^2(x) = k(x, x) - \mathbf{k}(x)^T (\mathbf{K} + \sigma_n^2 \mathbf{I})^{-1} \mathbf{k}(x) \quad (2)$$

where  $\mathbf{k}(x) = [k(x_1, x), \dots, k(x_K, x)]^T$  and  $\mathbf{K} = [k(x, x')]_{x, x' \in X}$

Clearly, using the above, we can compute the GP to update our knowledge about the unknown function  $f$  based on information acquired through observations.



## 2.2 Problem Definition

Given a GP and a domain  $X$ , we want to select a set of  $K$  points where to perform measurements in order to minimize the total posterior variance of the GP. Specifically we want to select a set  $K$  of measurements taken at locations  $\{x_1, x_2, \dots, x_K\}$  such that we minimize the following objective function:

$$J(K) = \sum_{x \in X} \sigma^2(x) \quad (3)$$

where  $\sigma^2(x)$  is computed using Eq. 2.

## 2.3 Submodularity

Define a set function as a function which inputs are sets of elements. Particular classes of set functions turn out to be submodular, which can be exploited in finding greedy solutions to optimization problems involving these types of functions. A fairly intuitive characterization of a submodular function has been given by Nemhauser et al. [9]: A function  $F$  is submodular if and only if for all  $A \subseteq B \subseteq X$  and  $x \in X \setminus B$  it holds that  $F(A \cup \{x\}) - F(A) \geq F(B \cup \{x\}) - F(B)$ .

The total posterior variance of a GP belongs to this class of functions, in which the set  $K$  of noisy measurements represents the input. Research in this context aimed at selecting such a set of measurement locations so as to minimize the posterior variance of the GP [6] and we mainly compare to this state-of-the-art method. Now, we are, in fact, going to exploit a much more direct method, which, surprisingly has not been studied in this context.

## 3 Gradient Descent Variance Reduction

Rather than exploiting the submodularity property of the objective function in Eq. 3 to come to a greedy subset selection, we decide to rely on standard GD. Specifically, starting from an initial configuration of measurement points in the domain, we perform a GD procedure to minimize the total posterior variance of the GP.

The main idea behind our algorithm is to exploit the gradient of the objective function in Eq. 3 to iteratively re-adapt the location of the measurements points across the domain. Notice that the value of the multi-dimensional objective function  $J(K)$  represents the total posterior variance of the GP given the  $K$  points in a  $d$  dimensional space. Following the gradient of the objective function corresponds to a simultaneous update of all the measurement points in the domain space. Considering these points simultaneously is what the submodular greedy approach does not do and what gives our approach an edge over that approach. In the direction of the negative gradient we have, in principle, a better solution and in our algorithm we take all the necessary precautions to avoid that the iterative step produces a displacement that would lead to a worse solution. With this, at every iteration the algorithm is guaranteed to obtain an improvement. A sketch of the pseudo-code is listed in Algorithm 1.

---

**Algorithm 1.** Gradient Descent (GD) procedure

**input:** set of initial sampling locations  $K_0$ , domain  $X$ , convergence factor  $cf$

---

```

1: Initialization
2: while not converged do
3:    $i \leftarrow i + 1$ ;  $step \leftarrow step + 1$ ;  $improved \leftarrow false$ 
4:   while not improved and not converged do
5:      $K_i \leftarrow K_{i-1} - \nabla J(K_{i-1})/step$ 
6:     if  $J(K_i) < J(K_{i-1})$  then
7:        $improved \leftarrow true$ 
8:     else
9:        $step \leftarrow step + 1$ ;  $K_i \leftarrow K_{i-1}$ 
10:    end if
11:    Check convergence using  $cf$ 
12:  end while
13: end while
14: return  $K_i$ 

```

---

Let us go through the procedure, starting out by describing the inputs and output that it considers. One of the inputs is the set of initial sampling points  $K$  that can be initialized using different choices. For example they can be chosen randomly or through use of a different techniques, a detailed description regarding our choices can be found in the experimental phase in Sect. 4. The second input, the domain  $X$ , represent the set of locations where we want to evaluate our GP in order to compute the posterior variance using Eq. 2. The remaining input ( $cf$ ) is used to determine the convergence of the procedure and it’s use will be clearer in the following description. The output of the procedure is represent by the final set  $K_i$  of sampling locations after  $i$  iteration of the algorithm.

The procedure begins with an initialization phase, here we initialize the required variables to manage the main loop and by computing the total posterior variance given the initial set of sampling locations  $K_0$ . The main loop (lines 2–13) iterates until the convergence is reached and it is made up of two main components: (i) the GD iterative step that allows to minimize the objective function (lines 4–12), described in Sect. 3.1; (ii) the check of convergence (line 11) whose function is described in Sect. 3.2.

### 3.1 Gradient Descent Iterative Step

Here we describe the function of the iterative step (lines 4–12) that allows our procedure to minimize the objective function. The iterative step computes for all the points in  $K$  (line 5) what is the new position given the derivative of the objective function. However, as any GD procedure, we have to keep into account situations where the iterative step would “jump” over the current basin of attraction. As noted earlier, in the direction of the negative gradient the objective function is decreasing in value and we want to guarantee that our algorithm at every iteration improves the solution. A simple method is to check whether the current step would make us improve the current solution or not. To this

aim we recompute the value of the objective function (line 6) and verify that this correspond to a net improvement with respect to the previous configuration. Otherwise we roll-back to the previous solution  $K_{i-1}$  and recompute a smaller displacement (line 9). To this aim we make use of the additional variable *step*. We can observe that this variable is used to compute the amplitude of the displacement in line 5. The *step* is increased at each iteration of the algorithm at least once (in line 3) to guarantee a slowdown, and an additional number of times (line 9) to guarantee that at each iteration we obtain an improvement (i.e. we minimize our objective function).

### 3.2 Convergence

As mentioned before, as part of the inputs we have *cf* which is used to determine the convergence of the algorithm. This parameter is intended as a threshold to determine whether the procedure has to terminate or not. *cf* specifies what is the lowest percentage (with respect of the dataset diameter) of displacement that any points we are adapting can move. At the beginning of the procedure (line 1) we also compute the diameter of the dataset, let's call it *maxD*. Inside the main loop of the procedure, we check the convergence (line 11). When all the points in  $K$  received a displacement that is lower than  $cf \cdot maxD$  we consider the procedure terminated. The *cf* parameter act as a trade-off between the precision of the solution and the computation (number of iteration) required to converge. For small values the algorithm is allowed to go through its iterations as long as at least one of the points in space is moving by a small amount. Larger values will make the procedure stop earlier with a solution that may of course be further from an optimum than when small values are used.

## 4 Dataset and Experimental Settings

To test the performance of our procedure under different conditions we generated datasets with domains in 1 to 5 dimensions. Specifically we have generated datasets with domain points  $X$  equally distributed over the dimensions. The cardinality of the domain  $|X|$ , that is the number of points on which we evaluate the GP, has been adapted to be at least 1000 points. The two dimensional dataset is simply a set of equally distributed points on a grid, while the three dimensional dataset is a set of equally distributed points on a cube, etc.

The most widely used kernel is Gaussian one (also known as squared exponential):  $K_{SE}(x, x') = \sigma_f^2 \exp\left(-\frac{(x-x')^2}{2l^2}\right)$  which is therefore the obvious choice in our experiments. The hyperparameters of the kernel can vary considerably however. Hence, to generally study the performance of our GD procedure we varied these in our experiments. Specifically we used 20 different length-scale  $l$  and 15 different  $\sigma_f$ . The former describes the smoothness property of the true underlying function while the latter the standard deviation of the modeled function. As we can observe in Eq. 2 these are fundamental to determine the variance

of the GP. Moreover, as mentioned in Sect. 2.1 we assume that measurements are noisy and in our experiments we also used 10 different  $\sigma_n$ .

In addition to the different number of dimensions of the datasets and the hyperparameters previously described, we have tested the procedure by adapting a different number of points (cardinality of the set  $K$ ) from 2 up to 7. The case of a single point has been excluded since the submodular greedy technique is optimal by definition.

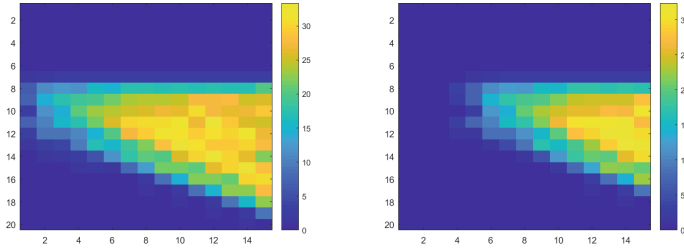
Some starting locations of the points are required to initialize our GD algorithm. Here we initialized them using the submodular greedy procedure in order to measure the magnitude of the possible improvements and to see under what conditions we can obtain them. The additional input of the procedure as described in Sect. 3 is  $cf = 1/1000$ .

To summarize, by considering the different hyperparameters, dimensionality of the datasets and number of measurement points, we have performed 90,000 different experiments that allows us to characterize and study the improvement obtainable with the GD procedure with respect to the widely used submodular greedy technique. Moreover, we also have performed the 90,000 experiments by initializing the points randomly instead of using a submodular solution, this allows us to study the average improvement obtainable without the needs to previously perform a different algorithm. In addition we have selected a subset of the hyperparameters and datasets to perform a test with many different random initialization on the same instances. The results of the experiments are described in the next section.

## 5 Results

We describe the results from different points of view and comment on the applicability of the technique we proposed. To explain the performance of GD as a function of the hyperparameters of the GP, we take as example the two plots in Fig. 1. In this pictures we can observe the % of improvement that GD obtains with respect to the submodular solution by varying the hyperparameters in the two dimensional dataset by adapting 5 points: vertically the length-scale  $l$  of the kernel and horizontally the standard deviation  $\sigma_f$  of the function. The two pictures represent these improvements by fixing a single standard deviation of the noise measurement  $\sigma_n$ ; the one to the right with a  $\sigma_n$  that is almost three times the one to the left.

To start with, independently of  $\sigma_f$  and  $\sigma_n$ , when we use very small length-scales (top rows of the two pictures) the advantage we can obtain with GD is very low. The reason why this happens is that with small length-scales the contribution in variance reduction given by an observations is mostly concentrated in a very narrow position. Consider that we are trying to estimate where to make two observations, as long as they are a little separated one another we are already obtaining most of the variance reduction possible. With very small length-scale the position where we make observations influences little to nothing the final amount of posterior variance. Hence with GD in these cases we cannot obtain an advantage with respect to the submodular greedy technique.



**Fig. 1.** Results as a function of the hyperparameters. Horizontally are variations in the standard deviation  $\sigma_f$  and vertically the length-scale  $l$ . Colors represent the % of variance reduction of GD relative to the submodular greedy solution. These results refer to 5 points in the 2-dimensional dataset and each picture for a fixed  $\sigma_n$ . Specifically in the right image  $\sigma_n$  is about three times higher then in the left one. (Color figure online)

Secondly, when the length-scale of the kernel becomes bigger the reduction in variance given by a measurement point has an effect on a larger portion of the domain, hence the location where the measurements are taken affect the total amount of posterior variance reduction. In this case we observe that the locations selected by the GD procedure obtain an advantage with respect to the submodular greedy technique.

Finally, when the length-scale becomes bigger we notice that the  $\sigma_f$  and  $\sigma_n$  parameters affect the results differently. Consider, for instance, the left picture in Fig. 1. The picture displays results for a fixed  $\sigma_n$ , with the other two variables on the two axes. We can observe that for small values of  $\sigma_f$  we obtain a small advantage and vice versa. These results are shifted to the right when the  $\sigma_n$  parameter increases (right picture in Fig. 1). This show that the ratio  $\sigma_f/\sigma_n$  affects the quality of the results: the higher the ratio the higher the improvements we can obtain.

### 5.1 Varying the Number of Points and Dimensionality

In this section we study the performance of GD with respect to the submodular greedy solution by varying the cardinality of the set  $K$  and the number of dimensions of the domain. In Table 1 we report the percentage of variance reduction that the GD procedure obtain with respect to the total posterior variance of the GP with the measurement locations selected with the submodular greedy technique. Specifically, each entry of the table reflects the improvement obtained for a specific combination of number of points and dimensionality of the domain.

Table 1 represents the average and maximum % gain of GD with respect to the submodular greedy solution. On the average columns each entry represents the average over all the 3000 hyperparameters for a specific combination of dimensionality of the domain and number of measurement points. As we can observe, in general the GD procedure allows us to improves significantly for small dimensionality and number of points. Regarding the maximum improvement

**Table 1.** Average and maximum % gain of GD with respect to the submodular solution

	Average improvement per number of points						Maximum improvement per number of points					
	2	3	4	5	6	7	2	3	4	5	6	7
1-D	32.8	18.2	17.6	17.1	14.8	8.5	59.9	86.8	89.8	89.2	71.6	71.7
2-D	4.1	16.9	19.7	9.2	13.7	14.5	21.1	60.3	54.9	33.4	76.7	72.3
3-D	1.0	2.8	8.8	8.0	10.6	8.2	6.2	15.8	52.1	29.9	41.2	31.0
4-D	0.3	1.0	1.9	5.1	3.5	4.9	6.6	11.5	12.2	31.1	20.7	22.6
5-D	0.0	0.6	1.1	1.7	3.9	2.2	3.0	8.8	8.2	17.5	40.1	22.6

each value reported is the maximum value encountered between all the possible 3000 combination of hyperparameters. Also in this case we can observe that GD produces better results for small dimensionality and number of points.

### 5.2 Random Initialization

Here we report the results similarly to the previous section. In this case the GD procedure has been initialized with points in randomly selected locations.

**Table 2.** Average and maximum % gain of GD with respect to a random configuration

	Average improvement per number of points						Maximum improvement per number of points					
	2	3	4	5	6	7	2	3	4	5	6	7
1-D	38.8	45.0	45.6	46.6	47.1	46.6	99.4	99.3	99.6	99.8	99.7	99.6
2-D	19.7	35.0	36.4	35.8	37.0	38.6	78.3	99.1	97.4	96.9	94.4	96.5
3-D	18.3	18.0	32.3	30.1	30.9	30.7	70.0	81.1	98.4	96.6	94.1	88.9
4-D	17.2	14.6	16.9	30.3	27.4	25.9	62.9	66.1	76.2	96.7	94.2	94.4
5-D	15.9	13.4	12.9	15.9	28.0	25.1	59.9	58.8	62.3	75.3	95.6	97.1

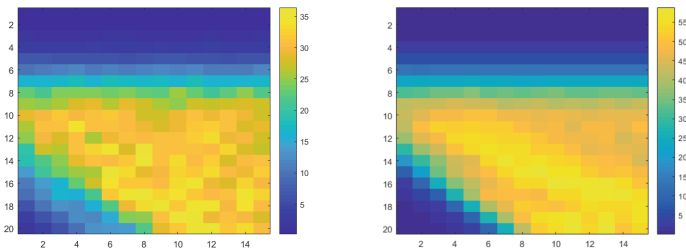
Table 2 represents the average and the maximum improvement of GD with respect to the random initial collocation of points. These results represent the gain in terms of percentage of variance reduction with respect to the variance of the GP with the measurement points in the random locations. Since the random collocation of points can represent a very bad quality solution compared to the submodular greedy procedure, results show much bigger improvements. A more interesting point of view is offered in Table 3. Here we compare the total posterior variance of the GP after the gradient descent adaptation from a random initialization with the total posterior variance after the gradient descent adaptation starting from the submodular greedy solution.

**Table 3.** Maximum % gain of gradient descent starting from a random configuration with respect to GD starting from the submodular greedy solution

	Number of points					
	2	3	4	5	6	7
1-D	43.4	76.0	74.0	39.1	53.2	36.9
2-D	14.2	34.6	31.9	35.3	52.1	52.1
3-D	9.7	15.8	30.2	16.4	35.9	21.9
4-D	4.9	7.7	14.1	26.6	15.3	15.3
5-D	1.2	7.0	7.0	7.2	26.7	21.4

Specifically, Table 3 reports the maximum improvements that have been encountered by varying the 3000 hyperparameters. Although, the result can vary considerably across the hyperparameters, results show that from a random initialization of points we can obtain in some cases better results than using a submodular greedy procedure to select the starting configuration.

Notice that the aforementioned Tables (2 and 3) report results considering a single random initialization per instance. Since the selection of the initial measurement points is subject to a great variance we also performed a more detailed test on a small subset of instances. Specifically, we have selected the 2-D dataset and we use gradient descent to adapt the location of two points and the 3-D dataset with six points. By fixing also a specific  $\sigma_n$  parameter, we performed experiments by using 100 randomly initialization for each of the 300 combinations of  $\sigma_f$  and  $l$ . Results are presented in Fig. 2. As we can observe, when we perform multiple randomly initialized executions on average we obtain a spectrum of improvements similar as what shown in previous Fig. 1.

**Fig. 2.** Average gain over 100 randomly initialized execution of GD. Left with 2 points in the 2-dimensional dataset and right 6 points in the 3-dimensional dataset.

## 6 Discussion and Conclusions

In this paper we proposed a Gradient Descent procedure to minimize the posterior variance of a GP. The performance of the technique has been analyzed

under different settings. Results show that in many cases it is possible to obtain a significant improvement with respect to a random or the well-known submodular greedy procedure. Although with a random initialization the performance can vary considerably, results show that in some cases it is possible to obtain better solutions than with a submodular greedy initialization.

It is also interesting to notice that in some applications, the locations where measurements are performed does not have to be confined in predetermined points in space, but rather the domain is continuous. Approaching this context by exploiting submodularity requires a discretization of the space. On the other hand GD does not requires the domain to be discrete and it can iteratively improve the solution by freely move the measurement points in a continuous manner.

Finally, GD is of course a general technique that can be applied to any differentiable objective function. It is therefore worthwhile to consider this technique in contexts where observations have to satisfy additional constraints, for example, when the points have to be confined to a specific region of the domain.

## References

1. Bottarelli, L., Bicego, M., Blum, J., Farinelli, A.: Skeleton-based orienteering for level set estimation. In: 22nd European Conference on Artificial Intelligence, ECAI 2016, Including Prestigious Applications of Artificial Intelligence, The Hague, The Netherlands, 29 August–2 September 2016, pp. 1256–1264 (2016)
2. Bottarelli, L., Blum, J., Bicego, M., Farinelli, A.: Path efficient level set estimation for mobile sensors. In: Proceedings of the Symposium on Applied Computing SAC 2017, pp. 262–267, ACM, New York, NY, USA (2017)
3. Guestrin, C., Krause, A., Singh, A.P.: Near-optimal sensor placements in Gaussian processes. In: Proceedings of the 22nd International Conference on Machine Learning, pp. 265–272. ACM (2005)
4. Krause, A., Guestrin, C.: Near-optimal observation selection using submodular functions. In: National Conference on Artificial Intelligence (AAAI), Nectar track, July 2007
5. Krause, A., Guestrin, C., Gupta, A., Kleinberg, J.: Robust sensor placements at informative and communication-efficient locations. *ACM Trans. Sen. Netw.* **7**(4), 31:1–31:33 (2011)
6. Krause, A., McMahan, H.B., Guestrin, C., Gupta, A.: Robust submodular observation selection. *J. Mach. Learn. Res.* **9**(Dec), 2761–2801 (2008)
7. Krause, A., Singh, A.: Near-optimal sensor placements in Gaussian processes: theory, efficient algorithms and empirical studies. *J. Mach. Learn. Res.* **9**(Feb), 235–284 (2008)
8. La, H.M., Sheng, W.: Distributed sensor fusion for scalar field mapping using mobile sensor networks. *IEEE Trans. Cybern.* **43**(2), 766–778 (2013)
9. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions–I. *Math. Program.* **14**(1), 265–294 (1978)
10. Powers, T., Bilmes, J., Krout, D.W., Atlas, L.: Constrained robust submodular sensor selection with applications to multistatic sonar arrays. In: 2016 19th International Conference on Information Fusion (FUSION), pp. 2179–2185, July 2016
11. Rasmussen, C.E., Williams, C.K.I.: *Gaussian Processes for Machine Learning*. MIT Press, Cambridge (2006)



# **Semi and Fully Supervised Learning Methods**



# Sparsification of Indefinite Learning Models

Frank-Michael Schleif<sup>1,2</sup>(✉), Christoph Raab<sup>1</sup>, and Peter Tino<sup>2</sup>

<sup>1</sup> Department of Computer Science,  
University of Applied Science Würzburg-Schweinfurt,  
97074 Würzburg, Germany  
{frank-michael.schleif, christoph.raab}@fhws.de  
<sup>2</sup> School of Computer Science, University of Birmingham,  
Birmingham B15 2TT, UK  
{schleify, p.tino}@cs.bham.ac.uk

**Abstract.** The recently proposed Krein space Support Vector Machine (KSVM) is an efficient classifier for indefinite learning problems, but with a non-sparse decision function. This very dense decision function prevents practical applications due to a costly out of sample extension. In this paper we provide a post processing technique to sparsify the obtained decision function of a Krein space SVM and variants thereof. We evaluate the influence of different levels of sparsity and employ a Nyström approach to address large scale problems. Experiments show that our algorithm is similar efficient as the non-sparse Krein space Support Vector Machine but with substantially lower costs, such that also large scale problems can be processed.

**Keywords:** Non-positive kernel · Krein space · Sparse model

## 1 Introduction

Learning of classification models for indefinite kernels received substantial interest with the advent of domain specific similarity measures. Indefinite kernels are a severe problem for most kernel based learning algorithms because classical mathematical assumptions such as positive definiteness, used in the underlying optimization frameworks are violated. As a consequence e.g. the classical Support Vector Machine (SVM) [24] has no longer a convex solution - in fact, most standard solvers will not even converge for this problem [9]. Researchers in the field of e.g. psychology [7], vision [17] and machine learning [2] have criticized the typical restriction to metric similarity measures. In fact in [2] it is shown that many real life problems are better addressed by e.g. kernel functions which are not restricted to be based on a metric. Non-metric measures (leading to kernels which are not positive semi-definite (non-psd)) are common in many disciplines. The use of divergence measures [20] is very popular for spectral data analysis in chemistry, geo- and medical sciences [11], and are in general not

metric. Also the popular Dynamic Time Warping (DTW) algorithm provides a non-metric alignment score which is often used as a proximity measure between two one-dimensional functions of different length. In image processing and shape retrieval indefinite proximities are often obtained by means of the inner distance [8] - another non-metric measure. Further prominent examples for genuine non-metric proximity measures can be found in the field of bioinformatics where classical sequence alignment algorithms (e.g. smith-waterman score [5]) produce non-metric proximity values. Multiple authors argue that the non-metric part of the data contains valuable information and should not be removed [17].

Furthermore, it has been shown [9, 18] that work-arounds such as eigen-spectrum modifications are often inappropriate or undesirable, due to a loss of information and problems with the out-of sample extension. A recent survey on indefinite learning is given in [18]. In [9] a stabilization approach was proposed to calculate a valid SVM model in the Krěin space which can be directly applied on indefinite kernel matrices. This approach has shown great promise in a number of learning problems but has intrinsically quadratic to cubic complexity and provides a dense decision model. The approach can also be used for the recently proposed indefinite Core Vector Machine (iCVM) [19] which has better scalability but still suffers from the dense model. The initial sparsification approach of the iCVM proposed in [19] is not always applicable and we will provide an alternative in this paper.

Another indefinite SVM formulation was provided in [1], but it is based on an empirical feature space technique, which changes the feature space representation. Additionally, the imposed input dimensionality scales with the number of input samples, which is unattractive in out of sample extensions.

The present paper improves the work of [19] by providing a sparsification approach such that the otherwise very dense decision model becomes sparse again. The new decision function approximates the original one with high accuracy and makes the application of the model practical.

The principle of sparsity constitutes a common paradigm in nature-inspired learning, as discussed e.g. in the seminal work [12]. Interestingly, apart from an improved complexity, sparsity can often serve as a catalyzer for the extraction of semantically meaningful entities from data. It is well known that the problem of finding smallest subsets of coefficients such that a set of linear equations can still be fulfilled constitutes an NP hard problem, being directly related to NP-complete subset selection. We now review the main parts of the Krěin space SVM provided in [9] showing why the obtained  $\alpha$ -vector is dense. The effect is the same for to the Core Vector Machine as shown in [19]. For details on the iCVM derivation we refer the reader to [19].

## 2 Krěin space SVM

The Krěin Space SVM (KSVM) [9], replaced the classical SVM minimization problem by a stabilization problem in the Krěin space. The respective equivalence between the stabilization problem and a standard convex optimization problem was shown in [9]. Let  $x_i \in X, i \in \{1, \dots, N\}$  be training points in the

input space  $X$ , with labels  $y_i \in \{-1, 1\}$ , representing the class of each point. The input space  $X$  is often considered to be  $\mathbb{R}^d$ , but can be any suitable space due to the kernel trick. For a given positive  $C$ , SVM is the minimum of the following regularized empirical risk functional.

$$J_C(f, b) = \min_{f \in \mathcal{H}, b \in \mathbb{R}} \frac{1}{2} \|f\|_{\mathcal{H}}^2 + CH(f, b) \quad (1)$$

$$H(f, b) = \sum_{i=1}^N \max(0, 1 - y_i(f(x_i) + b))$$

Using the solution of Equation (1) as  $(f_C^*, b_C^*) := \arg \min J_C(f, b)$  one can introduce  $\tau = H(f_C^*, b_C^*)$  and the respective convex quadratic program (QP)

$$\min_{f \in \mathcal{H}, b \in \mathbb{R}} \frac{1}{2} \|f\|_{\mathcal{H}}^2 \quad \text{s.t.} \quad \sum_{i=1}^N \max(0, 1 - y_i(f(x_i) + b)) \leq \tau \quad (2)$$

where we detail the notation in the following. This QP can be also seen as the problem of retrieving the orthogonal projection of the null function in a Hilbert space  $\mathcal{H}$  onto the convex feasible set. The view as a projection will help to link the original SVM formulation in the Hilbert space to a KSVM formulation in the Krein space. First we need a few definitions, widely following [9]. A Krěin space is an *indefinite* inner product space endowed with a Hilbertian topology.

**Definition 1 (Inner products and inner product space).** *Let  $\mathcal{K}$  be a real vector space. An inner product space with an indefinite inner product  $\langle \cdot, \cdot \rangle_{\mathcal{K}}$  on  $\mathcal{K}$  is a bi-linear form where all  $f, g, h \in \mathcal{K}$  and  $\alpha \in \mathbb{R}$  obey the following conditions: Symmetry:  $\langle f, g \rangle_{\mathcal{K}} = \langle g, f \rangle_{\mathcal{K}}$ , linearity:  $\langle \alpha f + g, h \rangle_{\mathcal{K}} = \alpha \langle f, h \rangle_{\mathcal{K}} + \langle g, h \rangle_{\mathcal{K}}$  and  $\langle f, g \rangle_{\mathcal{K}} = 0 \forall g \in \mathcal{K}$  implies  $f = 0$ .*

An inner product is positive definite if  $\forall f \in \mathcal{K}$ ,  $\langle f, f \rangle_{\mathcal{K}} \geq 0$ , negative definite if  $\forall f \in \mathcal{K}$ ,  $\langle f, f \rangle_{\mathcal{K}} \leq 0$ , otherwise it is indefinite. A vector space  $\mathcal{K}$  with inner product  $\langle \cdot, \cdot \rangle_{\mathcal{K}}$  is called inner product space.

**Definition 2 (Krěin space and pseudo Euclidean space).** *An inner product space  $(\mathcal{K}, \langle \cdot, \cdot \rangle_{\mathcal{K}})$  is a Krěin space if there exist two Hilbert spaces  $\mathcal{H}_+$  and  $\mathcal{H}_-$  spanning  $\mathcal{K}$  such that  $\forall f \in \mathcal{K}$ ,  $f = f_+ + f_-$  with  $f_+ \in \mathcal{H}_+$ ,  $f_- \in \mathcal{H}_-$  and  $\forall f, g \in \mathcal{K}$ ,  $\langle f, g \rangle_{\mathcal{K}} = \langle f_+, g_+ \rangle_{\mathcal{H}_+} - \langle f_-, g_- \rangle_{\mathcal{H}_-}$ . A finite-dimensional Krěin-space is a so called pseudo Euclidean space (pE).*

If  $\mathcal{H}_+$  and  $\mathcal{H}_-$  are reproducing kernel hilbert spaces (RKHS),  $\mathcal{K}$  is a reproducing kernel Krěin space (RKKS). For details on RKHS and RKKS see e.g. [15]. In this case the uniqueness of the functional decomposition (the nature of the RKHSs  $\mathcal{H}_+$  and  $\mathcal{H}_-$ ) is not guaranteed. In [13] the reproducing property is shown for a RKKS  $\mathcal{K}$ . There is a unique symmetric kernel  $k(x, x)$  with  $k(x, \cdot) \in \mathcal{K}$  such that the reproducing property holds (for all  $f \in \mathcal{K}$ ,  $f(x) = \langle f, k(x, \cdot) \rangle_{\mathcal{K}}$ ) and  $k = k_+ - k_-$  where  $k_+$  and  $k_-$  are the reproducing kernels of the RKHSs  $\mathcal{H}_+$  and  $\mathcal{H}_-$ .

As shown in [13] for any symmetric non-positive kernel  $k$  that can be decomposed as the difference of two positive kernels  $k_+$  and  $k_-$ , a RKKS can be

associated to it. In [9] it was shown how the classical SVM problem can be reformulated by means of a stabilization problem. This is necessary because a classical norm as used in Eq. (2) does not exist in the RKKS but instead the norm is reinterpreted as a projection which still holds in RKKS and is used as a regularization technique [9]. This allows to define SVM in RKKS (viewed as Hilbert space) as the orthogonal projection of the null element onto the set [9]:

$$S = \{f \in \mathcal{K}, b \in \mathbb{R} | H(f, b) \leq \tau\} \text{ and } 0 \in \partial_b H(f, b)$$

where  $\partial_b$  denotes the sub differential with respect to  $b$ . The set  $S$  leads to a unique solution for SVM in a Krěin space [9]. As detailed in [9] one finally obtains a stabilization problem which allows one to formulate an SVM in a Krěin space.

$$\text{stab}_{f \in \mathcal{K}, b \in \mathbb{R}} \frac{1}{2} \langle f, f \rangle_{\mathcal{K}} \quad \text{s.t.} \quad \sum_{i=1}^l \max(0, 1 - y_i(f(x_i) + b)) \leq \tau \quad (3)$$

where *stab* means stabilize as detailed in the following: In a classical SVM in RKHS the solution is regularized by minimizing the norm of the function  $f$ . In Krěin spaces however minimizing such a norm is meaningless since the dot-product contains both the positive and negative components. That's why the regularization in the original SVM through minimizing the norm  $f$  has to be transformed in the case of Krěin spaces into a min-max formulation, where we jointly minimize the positive part and maximize the negative part of the norm. The authors of [13] termed this operation the stabilization projection, or stabilization. Further mathematical details can also be found in [6]. An example illustrating the relations between minimum, maximum and the projection/stabilization problem in the Krěin space is illustrated in [9].

In [9] it is further shown that the stabilization problem Eq. (3) can be written as a minimization problem using a semi-definite kernel matrix. By defining a projection operator with transition matrices it is also shown how the dual RKKS problem for the SVM can be related to the dual in the RKHS. We refer the interested reader to [9]. One - finally - ends up with a flipping operator applied to the eigenvalues of the indefinite kernel matrix<sup>1</sup>  $K$  as well as to the  $\alpha$  parameters obtained from the stabilization problem in the Krěin space, which can be solved using classical optimization tools on the flipped kernel matrix. This permits to apply the obtained model from the Krěin space directly on the non-positive input kernel without any further modifications. The algorithm is shown in Algorithm 1. There are four major steps: (1) an eigen-decomposition of the full kernel matrix, with cubic costs (which can be potentially restricted to a few dominating eigenvalues - referred to as KSVM-L); (2) a flipping operation; (3) the solution of an SVM solver on the modified input matrix; (4) the application of the projection operator obtained from the eigen-decomposition on the  $\alpha$  vector of the SVM model.  $U$  in Algorithm 1 contains the eigenvectors,  $D$  is a diagonal matrix of the eigenvalues and  $S$  is a matrix containing only  $\{1, -1\}$  on the diagonal as obtained from the respective function sign.

<sup>1</sup> Obtained by evaluating  $k(x, y)$  for training points  $x, y$ .

---

**Algorithm 1.** Krěin Space SVM (KSVM) - adapted from [9].

---

**Krěin SVM:**  
 $[U, D] := \text{EigenDecomposition}(K)$   
 $\hat{K} := USDU^\top$  with  $S := \text{sign}(D)$   
 $[\alpha, b] := \text{SVMSolver}(\hat{K}, Y, C)$   
 $\tilde{\alpha} := USU^\top \alpha$  (now  $\tilde{\alpha}$  is dense)  
**return**  $\tilde{\alpha}, b;$

---

As pointed out in [9], this solver produces an exact solution for the stabilization problem. The main weakness of this Algorithm is, that it requires the user to pre-compute the whole kernel matrix and to decompose it into eigenvectors/eigenvalues. Further today's SVM solvers have a theoretical, worst case complexity of  $\approx \mathcal{O}(N^2)$ . The other point to mention is that the final solution  $\tilde{\alpha}$  is not sparse. The iCVM from [19] has a similar derivation and leads to a related decision function, again with a dense  $\tilde{\alpha}$ , but the model fitting costs are  $\approx \mathcal{O}(N)$ .

### 3 Sparsification of iCVM

#### 3.1 Sparsification of iCVM by OMP

We can formalize the objective to approximate the decision function, which is defined by the  $\tilde{\alpha}$  vector, obtained by KSVM or iCVM (both are structural identical), by a sparse alternative with the following mathematical problem:

$$\begin{aligned} \min \quad & |\tilde{\alpha}|_0 \\ \text{such that} \quad & \sum_m \tilde{\alpha}_m \Phi(x_m)^\top \Phi(x) \approx f(x) \end{aligned}$$

It is well-known that this problem is NP hard in general, and a variety of approximate solution strategies exist in the literature. Here, we rely on a popular and very efficient approximation offered by **orthogonal matching pursuit** (OMP) [3, 14]. Given an acceptable error  $\epsilon > 0$  or a maximum number  $n$  of non-vanishing components of the approximation, a greedy approach is taken: the algorithm iteratively determines the most relevant direction and the optimum coefficient for this axes to minimize the remaining residual error.

---

**Algorithm 2.** Orthogonal Matching Pursuit to approximate the  $\alpha$  vector.

---

1: **OMP:**  
2:  $I := \emptyset;$   
3:  $r := y := K\tilde{\alpha};$                    % initial residuum (evaluated decision function)  
4: **while**  $|I| < n$  **do**  
5:    $l_0 := \text{argmax}_l |[K r]_l|;$    % find most relevant direction + index  
6:    $I := I \cup \{l_0\}$                % track relevant indices  
7:    $\tilde{\gamma} := (K_{.I})^+ \cdot y$        % restricted (inverse) projection  
8:    $r := y - (K_{.I}) \cdot \tilde{\gamma}$        % residuum of the approximated decision function  
9: **end while**  
10: **return**  $\tilde{\gamma}$  (as the new sparse  $\tilde{\alpha}$ )

---

In line 3 of Algorithm 2 we define the initial residuum to be the vector  $K\tilde{\alpha}$  as part of the decision function. In line 5 we identify the most contributing dimension (assuming an empirical feature space representation of our kernel - it becomes the dictionary). Then in line 7 we find the current approximation of the sparse  $\tilde{\alpha}$ -vector - called  $\tilde{\gamma}$  to avoid confusion, where  $^+$  indicates the pseudo inverse. In line 8 we update the residuum by removing the approximated  $K\tilde{\alpha}$  from the original unapproximated one. A Nystroem based approximation of the Algorithm 2 is straight forward using the concepts provided in [4].

### 3.2 Sparsification of iCVM by Late Subsampling

The parameters  $\tilde{\alpha}$  are dense as already noticed in [9]. A naive sparsification by using only  $\tilde{\alpha}_i$  with large absolute magnitude is not possible as can be easily checked by counter examples. One may now approximate  $\tilde{\alpha}$  by using the (for this scenario slightly modified) OMP algorithm from the former section or by the following strategy, both compared in the experiments.

As a second sparsification strategy we can use the approach suggested by Tino et al. [19], to restrict the projection operator and hence the transformation matrix of iCVM to a subset of the original training data. We refer to this approach as ICVM-sparse-sub.

To get a consistent solution we have to recalculate parts of the eigen-decomposition as shown in Algorithm 3. To obtain the respective subset of the training data we use the samples which are core vectors<sup>2</sup>. The number of core vectors is guaranteed to be very small [22] and hence even for a larger number of classes the solution remains widely sparse. The suggested approach is given in Algorithm 3. We assume that the original projection function (line 6 of Algorithm 3, detailed in [9]), is smooth and can be potentially restricted to a small number of construction points with low error. We observed that in general few construction points are sufficient to keep high accuracy, as seen in the experiments.

---

#### Algorithm 3. Sparsification of iCVM by late subsampling

---

- 1: **Sparse iCVM:**
  - 2: Apply iCVM - see [19]
  - 3:  $\zeta$  - vector of projection points by using the core set points
  - 4: construct a reduced  $K'$  using indices  $\zeta$  as  $\bar{K}$
  - 5:  $[U,D] := \text{EigenDecomposition}(\bar{K})$
  - 6:  $\bar{\alpha} := USU^T\alpha$  with  $S := \text{sign}(D)$  and  $U$  restricted to the core set indices
  - 7:  $\tilde{\alpha} := 0$      $\tilde{\alpha}_\zeta := \bar{\alpha}$     % assign  $\bar{\alpha}$  to  $\tilde{\alpha}$  using indices of  $\zeta$
  - 8:  $b := Y\tilde{\alpha}^T$     % recalculate the bias using the (now) sparse  $\tilde{\alpha}$
  - 9: **return**  $\tilde{\alpha}, b$ ;
- 

<sup>2</sup> A similar strategy for KSVM may be possible but is much more complicated because typically quite many points are support vectors and special sparse SVM solvers would be necessary.

## 4 Experiments

This part contains a series of experiments that show that our approach leads to a substantially lower complexity, while keeping similar prediction accuracy compared to the non-sparse approach. To allow for large datasets with two much hassle we provide sparse results only for the iCVM. The modified OMP approach will work also for sparse KSVM but the late sampling sparsification is not well suited if many support vectors are given in the original model, asking for a sparse SVM implementation. We follow the experimental design given in [9]. Methods that require to modify test data are excluded as also done in [9]. Finally we compare the experimental complexity of the different solvers. The used data are explained in Table 1. Additional larger data sets have been added to motivate our approach in the line of learning with large scale indefinite kernels.

**Table 1.** Overview of the different datasets. We provide the dataset size ( $N$ ) and the origin of the indefiniteness. For vectorial data the indefiniteness is caused artificial by using the tanh kernel.

Dataset	#samples	Proximity measure and data source
Sonatas	1068	Normalized compression distance on midi files [18]
Delft	1500	Dynamic time warping [18]
a1a	1605	tanh kernel [10]
Zongker	2000	Template matching on handwritten digits [16]
Prodom	2604	Pairwise structural alignment on proteins [16]
PolydistH57	4000	Hausdorff distance [16]
Chromo	4200	Edit distance on chromosomes [16]
Mushrooms	8124	tanh kernel [21]
Swiss-10k	$\approx 10k$	Smith waterman alignment on protein sequences [18]
Checker-100k	100.000	tanh kernel (indefinite)
Skin	245.057	tanh kernel (indefinite)[23]
Checker	1 Mill	tanh kernel (indefinite)

### 4.1 Experimental Setting

For each dataset, we have run 20 times the following procedure: a random split to produce a training and a testing set, a 5-fold cross validation to tune each parameter (the number of parameters depending on the method) on the training set, and the evaluation on the testing set. If  $N > 1000$  we use  $m = 200$  randomly chosen landmarks from the given classes. If the input data are vectorial data we used a tanh kernel with parameters  $[1, 1]$  to obtain an indefinite kernel.



## 4.2 Results

Significant differences of iCVM to the best result are indicated by a  $\star$  (anova,  $p < 5\%$ ). In Table 2 we show the results for large scale data (having at least 1000 points) using iCVM with sparsification. We observe much smaller models, especially for larger datasets with often comparable prediction accuracy with respect to the non-sparse model. The runtimes are similar to the non-sparse case but in general slightly higher due to the extra eigen-decompositions on a reduce set of the data as shown in Algorithm 3.

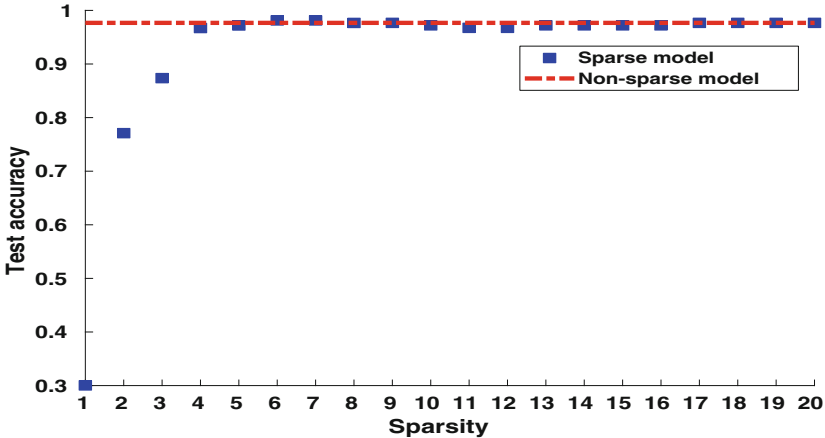
**Table 2.** Prediction errors on the test sets. The percentage of projection points (pts) is calculated using the unique set over core vectors over all classes in comparison to all training points. All sparse-OMP models use only 10 points in the final models. Best results are shown in bold. Best sparse results are underlined. Datasets with substantially reduced prediction accuracy are marked by  $\odot$ .

	iCVM (sparse-sub)	pts	iCVM (sparse-OMP)	iCVM (non-sparse)
Sonatas	<b><u>12.64 ± 1.71</u></b>	76.84%	22.56 ± 4.16 $\odot$	13.01 ± 3.82
Delft	16.53 ± 2.79 $\odot$	52.48%	<u>3.27 ± 0.6</u>	<b>3.20 ± 0.84</b>
a1a	39.50 ± 2.88 $\odot$	1.25%	<u>27.85 ± 2.8</u>	<b>20.56 ± 1.34</b>
Zongker	29.20 ± 2.48 $\odot$	52.81%	<u>7.50 ± 1.7</u>	<b>6.40 ± 2.11</b>
Prodom	<u>2.89 ± 1.17</u>	26.31%	3.12 ± 0.11	<b>0.87 ± 0.64</b>
PolydistH57	<u>6.12 ± 1.38</u>	12.92%	29.35 ± 8 $\odot$	<b>0.70 ± 0.19</b>
Chromo	11.50 ± 1.17	33.76%	<b><u>3.74 ± 0.58</u></b>	6.10 ± 0.63
Mushrooms	<u>7.84 ± 2.21</u>	6.46%	18.39 ± 5.7 $\odot$	<b>2.54 ± 0.56</b>
Swiss-10k	35.90 ± 2.52 $\odot$	17.03%	<b><u>6.73 ± 0.72</u></b>	12.08 ± 3.47
Checker-100k	<b>8.54 ± 2.35</b>	2.26%	19.54 ± 2.1 $\odot$	9.66 ± 2.32
Skin	<u>9.38 ± 3.30</u>	0.06%	9.43 ± 2.41	<b>4.22 ± 1.11</b>
Checker	8.94 ± 0.84	0.24%	<b><u>1.44 ± 0.3</u></b>	9.38 ± 2.73

A typical result for the protein data set using the OMP-sparsity technique and various values for sparsity is shown in Fig. 1.

## 4.3 Complexity Analysis

The original KSVM has runtime costs (with full eigen-decomposition) of  $\mathcal{O}(N^3)$  and memory storage  $\mathcal{O}(N^2)$ , where  $N$  is the number of points. The iCVM involves an extra Nyström approximation of the kernel matrix to obtain  $K_{(N,m)}$  and  $K_{(m,m)}^{-1}$ , if not already given. If we have  $m$  landmarks,  $m \ll N$ , this gives memory costs of  $\mathcal{O}(mN)$  for the first matrix and  $\mathcal{O}(m^3)$  for the second, due to the matrix inversion. Further a Nyström approximated eigendecomposition has to be done to apply the eigenspectrum flipping operator. This leads to runtime costs of  $\mathcal{O}(N \times m^2)$ . The runtime costs for the sparse iCVM are  $\mathcal{O}(N \times m^2)$  and the memory complexity is the same as for iCVM. Due to the used Nyström



**Fig. 1.** Prediction results for the protein dataset using a varying level of sparsity and the OMP sparsity methods. For comparison the prediction accuracy of the non-sparse model is shown by a straight line.

approximation the prior costs only hold if  $m \ll N$ , which is the case for many datasets as shown in the experiments.

The application of a new point to a KSVM or iCVM model requires the calculation of kernel similarities to all  $N$  training points, for the sparse iCVM this holds only in the worst case. In general the sparse iCVM provides a simpler out of sample extension as shown in Table 2, but is data dependent.

The (i) CVM model generation has not more than  $N$  iterations or even a constant number of 59 points, if the probabilistic sampling trick is used [22]. As show in [22] the classical CVM has runtime costs of  $\mathcal{O}(1/\epsilon^2)$ . The evaluation of a kernel function using the Nyström approximated kernel can be done with cost of  $\mathcal{O}(m^2)$  in contrast to constant costs if the full kernel is available. Accordingly, If we assume  $m \ll N$  the overall runtime and memory complexity of iCVM is linear in  $N$ , this is two magnitudes less as for KSVM for reasonable large  $N$  and for low rank input kernels.

## 5 Discussions and Conclusions

As discussed in [9], there is no good reason to enforce positive-definiteness in kernel methods. A very detailed discussion on reasons for using KSVM or iCVM is given in [9], explaining why a number of alternatives or pre-processing techniques are in general inappropriate. Our experimental results show that an appropriate Krěin space model provides very good prediction results and using one of the proposed sparsification strategies this can also be achieved for a sparse model in most cases. The proposed iCVM-sparse-OMP is only slightly better than the former iCVM-sparse-sub model with respect to the prediction accuracy but has

very few final modelling vectors, with an at least competitive prediction accuracy in the vast majority of data sets. As is the case for KSVM, the presented approach can be applied without the need for transformation of test points, which is a desirable property for practical applications. In future work we will analyse other indefinite kernel approaches like kernel regression and one-class classification.

**Acknowledgment.** We would like to thank Gaelle Bonnet-Loosli for providing support with the Krein Space SVM.

## References

1. Alabdulmohsin, I.M., Cissé, M., Gao, X., Zhang, X.: Large margin classification with indefinite similarities. *Mach. Learn.* **103**(2), 215–237 (2016)
2. Duin, R.P.W., Pekalska, E.: Non-euclidean dissimilarities: causes and informativeness. In: Hancock, E.R., Wilson, R.C., Windeatt, T., Ulusoy, I., Escolano, F. (eds.) *SSPR /SPR. LNCS*, vol. 6218, pp. 324–333. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-14980-1\\_31](https://doi.org/10.1007/978-3-642-14980-1_31)
3. Geoffrey, Z.Z., Davis, M., Mallat, S.G.: Adaptive time-frequency decompositions. *SPIE J. Opt. Eng.* **33**(1), 2183–2191 (1994)
4. Gisbrecht, A., Schleif, F.-M.: Metric and non-metric proximity transformations at linear costs. *Neurocomputing* **167**, 643–657 (2015)
5. Gusfield, D.: *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, Cambridge (1997)
6. Hassibi, B.: Indefinite metric spaces in estimation, control and adaptive filtering. Ph.D. thesis, Stanford University, Department of Electrical Engineering, Stanford (1996)
7. Hodgetts, C.J., Hahn, U.: Similarity-based asymmetries in perceptual matching. *Acta Psychol.* **139**(2), 291–299 (2012)
8. Ling, H., Jacobs, D.W.: Shape classification using the inner-distance. *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(2), 286–299 (2007)
9. Loosli, G., Canu, S., Ong, C.S.: Learning SVM in Krein spaces. *IEEE Trans. Pattern Anal. Mach. Intell.* **38**(6), 1204–1216 (2016)
10. Luss, R., d’Aspremont, A.: Support vector machine classification with indefinite kernels. *Math. Program. Comput.* **1**(2–3), 97–118 (2009)
11. Mwebaze, E., Schneider, P., Schleif, F.-M., et al.: Divergence based classification in learning vector quantization. *Neurocomputing* **74**, 1429–1435 (2010)
12. Olshausen, B.A., Field, D.J.: Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vis. Res.* **37**(23), 3311–3325 (1997)
13. Ong, C.S., Mary, X., Canu, S., Smola, A.J.: Learning with non-positive kernels. In: *(ICML 2004)* (2004)
14. Pati, Y.C., Rezaifar, R., Krishnaprasad, P.S.: Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In *Proceedings of the 27th Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 40–44, November 1993
15. Pekalska, E., Duin, R.: *The Dissimilarity Representation for Pattern Recognition*. World Scientific, Singapore (2005)
16. Pekalska, E., Haasdonk, B.: Kernel discriminant analysis for positive definite and indefinite kernels. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(6), 1017–1031 (2009)

17. Scheirer, W.J., Wilber, M.J., Eckmann, M., Boulton, T.E.: Good recognition is non-metric. *Pattern Recogn.* **47**(8), 2721–2731 (2014)
18. Schleif, F.-M., Tiño, P.: Indefinite proximity learning: a review. *Neural Comput.* **27**(10), 2039–2096 (2015)
19. Schleif, F.-M., Tiño, P.: Indefinite core vector machine. *Pattern Recogn.* **71**, 187–195 (2017)
20. Schnitzer, D., Flexer, A., Widmer, G.: A fast audio similarity retrieval method for millions of music tracks. *Multimed. Tools Appl.* **58**(1), 23–40 (2012)
21. Srisuphab, A., Mitrpanont, J.L.: Gaussian kernel approx algorithm for feedforward neural network design. *Appl. Math. Comp.* **215**(7), 2686–2693 (2009)
22. Tsang, I.H., Kwok, J.Y., Zurada, J.M.: Generalized core vector machines. *IEEE TNN* **17**(5), 1126–1140 (2006)
23. UCI: Skin segmentation database, March 2016
24. Vapnik, V.N.: *The Nature of Statistical Learning Theory*. Statistics for Engineering and Information Science. Springer, New York (2000)



# Semi-supervised Clustering Framework Based on Active Learning for Real Data

Ryosuke Odate<sup>(✉)</sup>, Hiroshi Shinjo, Yasufumi Suzuki,  
and Masahiro Motobayashi

Hitachi Ltd. Research and Development Group,  
1-280, Higashi-koigakubo, Kokubunji-shi, Tokyo 185-8601, Japan  
ryosuke.odate.qs@hitachi.com

**Abstract.** In this paper, we propose a real data clustering method based on active learning. Clustering methods are difficult to apply to real data for two reasons. First, real data may include outliers that adversely affect clustering. Second, the clustering parameters such as the number of clusters cannot be made constant because the number of classes of real data may increase as time goes by. To solve the first problem, we focus on labeling outliers. Therefore, we develop a stream-based active learning framework for clustering. The active learning framework enables us to label the outliers intensively. To solve the second problem, we also develop an algorithm to automatically set clustering parameters. This algorithm can automatically set the clustering parameters with some labeled samples. The experimental results show that our method can deal with the problems mentioned above better than the conventional clustering methods.

**Keywords:** Clustering · Semi-supervised · Real data  
Automatic parameters setting · Stream based · Active learning  
Ward's method · Classification

## 1 Introduction

Clustering has been widely used for data analysis [1–3]. The usages of clustering are roughly divided into two types [4].

The first usage is data trend analysis. Since data trend analysis by clustering is unsupervised learning, people need to subjectively decide how to divide clusters. People supplementarily use the clustering results for summarizing data and acquiring knowledge. Thus, there are no correct or incorrect results in the data trend analysis by clustering.

The second usage is data classification. Since the clustering is unsupervised learning, it cannot be used for classification directly. However, for data with objective classification criteria, we can use clustering methods to derive the classifier. In the research area of classification using clustering, semi-supervised

clustering has been studied [5–7]. This approach can create a classifier from the clustering results on unlabeled data by introducing a small amount of labeled data and clustering constraints [8]. Although researchers often use supervised learning methods such as learning vector quantization [9] for classification problems, these methods are not designed to classify unlabeled data. Semi-supervised clustering is a good approach to classify unlabeled data.

Since the utilization of big data has become common, demand for real data analysis has been increasing. In this paper, we define real data as unprocessed data for machine learning; that is, real data includes outliers and errors. In addition, real data is not always labeled, and the number of the classes is not always counted. For example, the raw data acquired by sensors is real data. Such data exists in various environments and is accumulated every day in factories, hospitals, and so on.

Semi-supervised clustering is suitable for real data classification because real data is often unlabeled or sparsely labeled. However, the conventional semi-supervised clustering methods are difficult to apply to real data directly for two reasons. First, real data includes outliers and errors. If we use a conventional method with such samples, the cluster to be divided may be mixed. Second, the number of clusters and the thresholds of cluster division cannot be set to be constant because the number of classes of real data may increase as time goes by. In this paper, we consider the number of clusters and clustering threshold as clustering parameters. When people use conventional clustering methods, they usually decide clustering parameters in advance. For example, if we use  $k$ -means [10], we have to decide the number of clusters  $k$  in advance. In contrast, when we apply clustering methods to real data, we cannot decide  $k$  in advance. Furthermore, we have to decide  $k$  whenever the number of classes increases.

In this paper, we propose a semi-supervised clustering framework based on active learning for real data. We address a very specific type of semi-supervised clustering, namely, working with hard cluster assignments. This excludes techniques such as Gaussian mixture models [11] and fuzzy clustering techniques [12]. Generally, active learning selects the unlabeled samples and then requests annotators to label the samples. The annotator is a human who provides the correct label. This technique is often used to have classifiers learn effectively with few labeled samples. In our method, we use this technique to label outliers and errors intensively. We introduce active learning [13] to Ward's method [14] as an example in this paper but also propose a framework. Therefore, Ward's method is compatible with the other clustering methods. We also develop an algorithm to automatically set clustering parameters. This algorithm automatically updates parameters in response to increases in the number of samples and clusters.

The rest of this paper is organized as follows. Section 2 clarifies the problem of real data clustering. We then present our approach to solve those problems in Sect. 3. In Sect. 4, we propose a clustering method based on active learning. Section 5 describes the experimental results and discussions, and Sect. 6 concludes this paper.

## 2 Problem Settings

### 2.1 Real Data Clustering

We use clustering methods for classification. Figure 1(b) is a schematic diagram of clustering results. Hence, we can consider Fig. 1(b) as a schematic diagram of a classifier made by clustering. If the input belongs to one of the clusters, the input can be classified as a specific class. Therefore, when the condition

$$\text{input} \in c_i \quad (1 \leq i \leq \text{the number of clusters}) \quad (1)$$

is satisfied, input is classified as cluster  $i$ .  $c_i$  is a cluster created by clustering on learning data. Each cluster should contain only one class of learning data. Our method is one of the hard clustering methods. Therefore, our task are different from that of the conventional methods that allow ambiguity [11,12].

There are two main problems in classification by clustering in real data.

1. Outliers and errors
2. Changes in the number of samples and clusters.

Both problems cause abnormalities in the number of clusters and the number of classes in a cluster. We describe each problem in detail in the next subsection.

### 2.2 Problem 1: Outliers and Errors

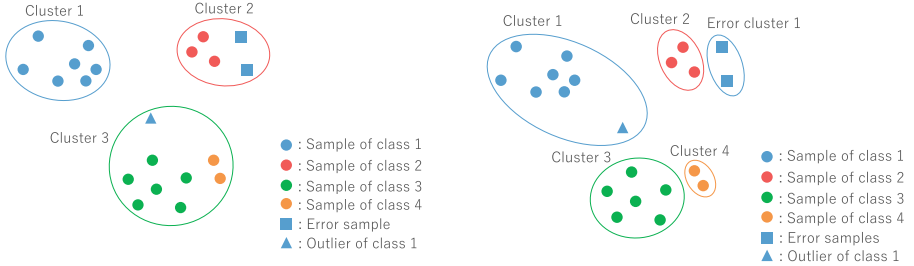
Outliers and errors rarely exist in the data processed for machine learning but exist in real data. For example, errors may be acquired because the sensor malfunctions or the measurement environment is different from usual by chance.

Figure 1(a) shows a schematic diagram when we try to divide learning data into three clusters using the conventional unsupervised clustering method. To clarify that the clustering result is wrong, correct labels are given to the samples in this figure. Assuming that the clustering results such as those in Fig. 1(a) are a classifier, the classifier identifies the class of an input sample by checking which cluster contains the input sample. Therefore, for classification, each cluster should consist of the samples of only one class. However, outliers and errors cause clustering mistakes. Explaining this more specifically with reference to the figure, cluster 2 in Fig. 1(a) includes errors that should not be included and therefore is expanded by errors. Second, cluster 3 is expanded by the outlier of class 1. In the case of such a classifier, the input satisfies Eq. (1) with an incorrect cluster. As a result, incorrect classification occurs.

### 2.3 Problem 2: Change in Number of Samples and Clusters

The number of samples of real data may increase as time goes by. Furthermore, the number of classes of real data may increase. Since many conventional clustering methods target the data whose classes do not increase, they have difficulty dealing with real data. Figure 1(a) shows the case where three-class classification

was assumed but a fourth class appeared. In this case, class 4 is forced into cluster 3. If we use clustering to analyze data trends, the clustering results are not a problem. The reason is that clustering is only analyzing data subjectively to divide it into three classes. However, if we use clustering to classify samples, the results are a problem. The classifier learns erroneously every time the number of classes increases.



(a) Incorrect clustering results for outliers, errors, and samples of a new class.

(b) Ideal clustering results.

**Fig. 1.** Schematic diagram of clustering

## 3 Approach

### 3.1 Overview

The ideal clustering results are shown in Fig. 1(b). All clusters consist of samples of one class in this figure. To obtain this result, we need to solve the two problems mentioned in Sect. 2. We thus introduce two approaches.

1. Stream Based Active Learning
2. Automatic Parameter Setting.

To solve problem 1 (Sect. 2.2), we label outliers and errors with stream based active Learning. In addition, to solve problem 2 (Sect. 2.3), the classifier should automatically set clustering parameters as samples increase. We define clustering parameters as the number of clusters and the threshold of cluster division. The following subsections present approaches in detail with reference to Fig. 1.

### 3.2 Stream Based Active Learning

In this paper, the annotator is a human. The annotators label the samples not satisfying Eq. (1) to incorporate these samples into learning as teaching data. The samples that does not satisfy Eq. (1) are regarded as outliers or errors at that time. We introduce stream based active learning into clustering. This algorithm contributes to labeling outliers and errors intensively with less effort. If the annotators label a sample that does not belong to any clusters, the classifier



can learn whether the sample is an error, an outlier of an existing class, or the sample of a new class.

Active learning is a method to select samples effective for a learning classifier and request annotators to label them. A stream based method [15,16] can deal with the data that may increase as time goes by. Real data is not pooled; it is a stream.

Referring to Fig. 1(a), we assume that clusters 1 and 2 are formed and cluster 3 is not. Then if the triangular sample is input there, it should be labeled “Outlier of class 1” and incorporated into cluster 1 as in Fig. 1(b).

### 3.3 Automatic Parameter Setting

Since samples not in any clusters are labeled by active learning as described in Sect. 3.2, an algorithm is needed to set clustering parameters automatically by using the labeled samples. This is a semi-supervised clustering-like approach. The contribution of this algorithm is that parameter setting by a person is unnecessary. As a result, this algorithm makes it easy to introduce clustering methods because parameter setting based on domain knowledge will be unnecessary. In this approach, each cluster has the individual threshold of a cluster division. The individual threshold allows us to extend only one cluster with large variance such as cluster 1 in Fig. 1(b).

Referring to Fig. 1(b), if the center sample is labeled “Outlier of class 1”, set the clustering parameters to expand cluster 1. If the upper right samples are labeled “Error of class 1”, generate “Error cluster 1”, i.e. generate a new class “Error 1”. If the bottom right samples are labeled “Class 4”, generate a new cluster, “Cluster 4”.

In this way, the algorithm automatically decides the parameters that people have to decide normally. In other words, this algorithm makes classifiers re-learned when unclassifiable samples are input. If a sample similar to such unclassifiable samples is input next time, the classifier will be able to classify it.

## 4 Proposed Method

### 4.1 Overview

In this section, we describe the details of our method, a semi-supervised clustering framework based on active learning. This method is based on the approaches introduced in Sect. 3.

First, this subsection briefly presents the outline of the proposed method. The proposed method consists of three algorithms: classification, active learning, automatic parameter setting. Since the classifier can be converted into an arbitrary clustering method, our proposed method is a framework.

It starts when a new sample is entered. To classify a new sample, a clustering method is used (Classification). If the new sample belongs to one of the existing clusters, the classification is completed. On the other hand, if the new sample

does not belong to any clusters, the sample is an error or outlier. Thus, the sample is labeled by active learning (Active learning). Thereafter, the clustering parameters are re-learned (Automatic parameter setting). This is one loop. We continue the loop as long as a new sample enters.

## 4.2 Classification

We use a conventional clustering method for the classification. Monotonic clustering methods are suitable for our method because the inclusion relationship between clusters is clear in their clustering results. For that reason, we chose Ward's method [14], which is a monotonic and hierarchical clustering method. This method joins two clusters in a bottom-up manner. Ward's method selects two clusters and joins them so as to minimize the value of the following equation.

$$d(c_1, c_2) = Var(c_1 \cup c_2) - (Var(c_1) + Var(c_2)) \quad (2)$$

$d(C_1, c_2)$  is the distance between clusters  $c_1$  and  $c_2$ .  $Var(c_1)$  and  $Var(c_2)$  are variance in clusters  $c_1$  and  $c_2$ . Ward's method is only one example of a clustering algorithm, and other hierarchical clustering methods can be also used. Since we use Ward's method with variance, we assume Gaussian distribution implicitly for each class in classification. However, since this method separates outliers as new classes (Fig. 1(b)), we do not forcefully assume Gaussian distribution on all samples in each class.

## 4.3 Stream Based Active Learning

Algorithm 1 shows the details of stream based active learning. Since active learning involves all processes of our method, Algorithm 1 contains almost all the details of our entire method.

With reference to Algorithm 1, we describe the learning process. In this algorithm, input is a dataset  $X$ .  $N_X$  is the number of samples and increases as time goes by. Output is a request to label  $x_i$  for the annotator. First, Ward's method is used to obtain a dendrogram  $D$  representing a cluster configuration. Second, labeled samples are collected and become labeled dataset  $X_L$ . Third, classifier  $G$  is trained by using Algorithm 2. At this time,  $G$  learns with dataset  $X_L$  labeled in the previous loop. After that, the samples of dataset  $X$  are classified using classifier  $G$ . A labeling request is presented to a sample that does not belong to any clusters  $C = \{c_i\}_{i=1}^{N_C}$ . This algorithm continues to run until there is no more input. The more the algorithm loops, the more accurate the classification.

## 4.4 Automatic Parameters Setting

Algorithm 2 shows the details of automatic parameter setting. This is an algorithm to learn a classifier using labeled data added by the active learning algorithm in Sect. 4.3.

---

**Algorithm 1.** Stream based active learning

---

```

Input :  $X = \{x_i\}_{i=1}^{N_X}$ 
Output: request annotators to label  $x_i$ 
1 while  $User\_stop = False$  do // continue until  $N_X$  does not increase
2    $D = Ward's\_method(X)$  // use Ward's method with  $X$ 
3   for  $i$  in  $range(N_x)$  do
4     if  $x_i$  is labeled then // make labeled dataset  $X_L$ 
5        $add\ x_i\ to\ X_L$ 
6     end
7   end
8    $G = Algorithm\ 2(D, X_L)$  // train classifier  $G$  by Algorithm 2
9   classify X by using G // determine to which cluster  $c_j$   $x_i$  belongs
10  if exists  $x_i \notin C$  then // clusters  $C = \{c_j\}_{j=1}^{N_C}$ 
11     $request\ annotators\ to\ label\ x_i$ 
12  end
13  stop // stop until a new sample is entered
14  if  $N_X$  increase then
15    start
16  end
17 end

```

---

In this algorithm, input is a dendrogram  $D$  and a labeled dataset  $X_L$ .  $N_{X_L}$  is the number of labeled samples. Output is a trained classifier  $G$ . This algorithm repeats the matching of labels of two or more samples falling into the same node  $s_i$  in the dendrogram  $D$ .  $S = \{s_j\}_{j=1}^{N_C}$  is the nodes of  $D$ .  $N_C$  is the number of the nodes  $S$ . In other words,  $S$  is a cluster's candidates and  $N_C$  is the number of the cluster's candidates. If the labels of the matching samples are the same, a cluster containing those samples is built. Then the division threshold of the cluster is updated to a value for including matching samples.

## 5 Experimental Results and Discussions

### 5.1 Datasets

We use three datasets from the UCI Machine Learning Repository [17]: Iris, Ecoli, and Leaf. The composition of each dataset is listed in Table 1. The same experiment is performed for each dataset.

In this experiment, we do not divide the dataset into learning and testing. We randomly rearranged each dataset and continue to input samples one by one into the classifier as in reality. Therefore, the data entered when the classifier is immature is used for learning. For example, learning outliers to extend clusters, learning errors to generate new clusters. On the other hand, the data entered when the classifier is mature is used for testing.

**Algorithm 2.** Automatic parameters setting

---

**Input** :  $D, X_L = \{x_{Li}\}_{i=1}^{N_{XL}}$   
**Output**:  $G$

```

1 count  $N_C$   $k \leftarrow 0$ 
2 for  $j$  in  $range(N_C)$  do
3   if exists two or more  $x_L \in S_j$  then // check existence of labeled data
4     if  $x_{LS} \in S_j$  are the same labeled then
5       construct  $c_k$  from  $x_{LS} \in S_j$  // construct a larger cluster
6        $T_k =$  distance between  $x_{LS} \in S_j$  // set the threshold
7       register  $c_k$  and  $T_k$  with  $G$  // construct a classifier
8        $k \leftarrow k + 1$ 
9     end
10  end
11 end
12 return  $G$ 

```

---

**Table 1.** Datasets.

Dataset	Iris	Ecoli	Leaf
Samples	150	336	340
Class	3	8	36
Attribute	4	8	16

**5.2 Performance Evaluation on UCI Machine Learning Datasets**

We evaluate proposed method in the two viewpoints. The first is the number of labeled samples. In this experiment, since all data is regarded as unlabeled and input, the number of labeled samples leads to operational cost. The second is the accuracy of classification expressed by the following equation.

$$Accuracy = \frac{\text{correctly classified samples}}{\text{all samples} - \text{labeled samples}} \quad (3)$$

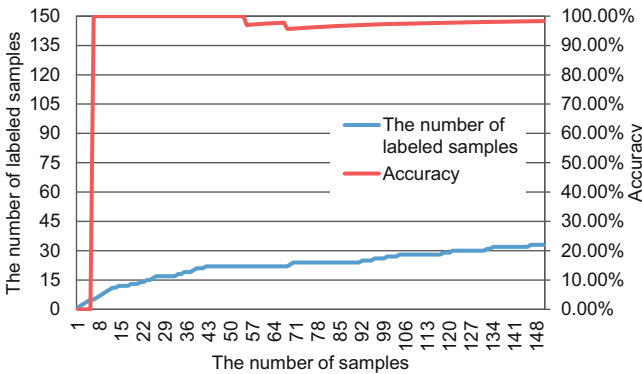
We show the performance after inputting all samples on each dataset in Table 2. By labeling with active learning, the accuracy can be maintained while responding to the increase in the number of classes. The accuracy is especially high in the Iris dataset: 98.29% because the Iris dataset contains many linearly separable samples. We labeled more samples in the Leaf dataset than the Iris because the Leaf datasets have many classes and samples that are difficult to linearly separate. Since the conventional method cannot cope with the increase in the number of classes, it cannot be compared with the proposed method.

Figure 2 shows the accuracy and the number of labeled samples in the Iris dataset. The number of labeled samples first increases linearly and gradually saturates. Although the accuracy is basically high, this method misclassified two

samples. The Iris dataset consists of three classes. Although one class is separated, the other two are partly mixed in the feature space. The misclassification occurred on these partly mixed samples. This tendency is the same in the other datasets. Therefore, our method is the best at classifying data that can be linearly separated in the feature space. In addition, in this case, fewer labels are required. As long as linear separation is possible, it seems that classification can be done with less labeling cost no matter how much classes are increased. To extend the application targets in the future, it is necessary to extract linearly separable features or introduce classifiers capable of nonlinear classification. In this case, the proposed framework can also be used.

**Table 2.** Number of labeled samples and accuracy after inputting all samples on each dataset.

Dataset	Iris	Ecoli	Leaf
Labeled samples	33	98	199
Accuracy [%]	98.29	90.34	88.65



**Fig. 2.** Number of labeled samples and accuracy involved in increase of learning data.

## 6 Conclusions

This paper has presented a real data clustering method based on active learning. We have introduced active learning into Ward's method. This technique makes clustering robust against outliers. In addition, we developed an automatic parameter setting algorithm. This algorithm automatically sets parameters as the number of classes changes. This enables our clustering method to cope with the change in the number of classes without people setting the parameters. The experimental results show that our method can deal with outliers and changes

in the number of classes. In the Iris dataset, we constructed a classifier that achieves 98.29% classification accuracy when labeling 33 samples.

For future work, we aim to use another clustering method for a classifier and to extend the application targets.

## References

1. Halim, Z., Atif, M., Rashid, A.: Profiling players using real-world datasets: clustering the data and correlating the results with the big-five personality traits. *IEEE Trans. Affect. Comput.*, 1–18 (2017)
2. Bijuraj, L.V.: Clustering and its applications. In: *Proceedings of National Conference on New Horizons in IT - NCNHIT 2013*, pp. 169–172 (2013)
3. Tran, N., Vo, B., Phung, D.: Clustering for point pattern data. In: *Proceedings of the 2016 23rd International Conference on Pattern Recognition (2013)*
4. Kamishima, T., Motoyoshi, F.: Learning from cluster examples. *Mach. Learn.* **53**(3), 199–233 (2003)
5. Bair, E.: Semi-supervised clustering methods. *Wiley Interdisc. Rev. Comput. Stat.* **5**(5), 349–361 (2013)
6. Grira, N., Crucianu, M., Boujemaa, N.: Unsupervised and semi-supervised clustering: a brief survey. In: *Proceedings of the Review of Machine Learning Techniques for Processing MUSCLE European Network of Excellence (2004)*
7. Wang, Y., Chen, S., Zhou, Z.: New semi-supervised classification method based on modified cluster assumption. *IEEE Trans. Neural Netw. Learn. Syst.* **23**(5), 689–702 (2012)
8. Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained k-means clustering with background knowledge. In: *Proceedings of the 9th ICML*, pp. 577–584 (2001)
9. Kohonen, T.: *Self-Organizing Maps*, vol. 30. Springer, Heidelberg (2001). <https://doi.org/10.1007/978-3-642-56927-2>
10. Macqueen, J.: Some methods for classification and analysis of multivariate observations. In: *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297 (1967)
11. Martinez-Usó, A., Pla, F., Sotoca, J.: A semi-supervised Gaussian mixture model for image segmentation. In: *Proceedings of 20th International Conference on Pattern Recognition*, pp. 2941–2944 (2010)
12. Grira, N., Crucianu, M., Boujemaa, N.: Active semi-supervised fuzzy clustering. *Pattern Recogn.* **41**(5), 1834–1844 (2008)
13. Gosselin, P.H., Cord, M.: Active learning methods for interactive image retrieval. *IEEE Trans. Image Process.* **17**(7), 1200–1211 (2008)
14. Ward Jr., J.H.: Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.* **58**(301), 236–244 (1963)
15. Narr, A., Triebel, R., Cremers, D.: Stream-based active learning for efficient and adaptive classification of 3D objects. In: *Proceedings of 2016 IEEE International Conference on Robotics and Automation (2016)*
16. Fujii, K., Kashima, H.: Budgeted stream-based active learning via adaptive sub-modular maximization. In: *Proceedings of Conference and Workshop on Neural Information Processing Systems (2016)*
17. Dua, D., Karra Taniskidou, E.: UCI machine learning repository (2017). <http://archive.ics.uci.edu/ml&gt>



# Supervised Classification Using Feature Space Partitioning

Ventzeslav Valev<sup>1</sup>, Nicola Yanev<sup>1</sup>, Adam Krzyżak<sup>2</sup>(✉),  
and Karima Ben Suliman<sup>2</sup>

<sup>1</sup> Institute of Mathematics and Informatics, Bulgarian Academy of Sciences,  
Sofia, Bulgaria

{valev, choby}@math.bas.bg

<sup>2</sup> Department of Computer Science and Software Engineering, Concordia University,  
Montreal, Quebec H3G 1M8, Canada

krzyzak@cs.concordia.ca, karima.b.soliman@gmail.com

**Abstract.** In the paper we consider the supervised classification problem using feature space partitioning. We first apply heuristic algorithm for partitioning a graph into a minimal number of cliques and subsequently the cliques are merged by means of the nearest neighbor rule. The main advantage of the new approach which optimally utilizes the geometrical structure of the training set is decomposition of the  $l$ -class problem ( $l > 2$ ) into  $l$  single-class optimization problems. We discuss computational complexity of the proposed method and the resulting classification rules. The experiments in which we compared the box algorithm and SVM show that in most cases the box algorithm performs better than SVM.

**Keywords:** Supervised classification · Feature space partitioning  
Graph partitioning · Nearest neighbor rule · Box algorithm

## 1 Introduction

This paper considers the supervised classification problem in which a pattern is assigned to one of a finite number of classes. The goal of supervised classification is to learn a function,  $f(x)$  that maps features  $x \in X$  to a discrete label (color),  $y \in \{1, 2, \dots, l\}$  based on training data  $(x_i, y_i)$ . Our proposal is to approximate  $f$  by partitioning the feature space into uni-colored box-like regions. The optimization problem of finding the minimal number of such regions is reduced to the well-known problem of minimum clique cover of a properly constructed graph. The solution results in feature space partitioning. This geometrical approach has been recently actively pursued in the literature. We provide a brief survey of relevant results.

Many important intractable problems are easily reducible to minimum number of the Maximum Clique Problem (MCP), where the Maximal Clique is the largest subset of vertices such that each vertex is connected to every other vertex in the subset. They include the Boolean satisfiability problem, the

independent set problem, the subgraph isomorphism problem, and the vertex covering problem.

In the literature much attention has been devoted to developing efficient heuristic approaches for MCP for which no formal guarantee of performance exist. These approaches are nevertheless useful in practical applications. In [1] a flexible annealing chaotic neural network has been introduced, which on graphs from the Center for Discrete Mathematics and Theoretical Computer Science (DIMACS) has achieved optimal or near-optimal solution. In [2] the proposed learning algorithm of the Hopfield neural network has two phases: the Hopfield network updating phase and the gradient-ascent learning phase. In [3] annealing procedure is applied in order to avoid local optima. Another algorithm for MCP on arbitrary undirected graph is described in [4]. The algorithm presumes that vertices from an independent set (i.e. a set of vertices that are pairwise nonadjacent) cannot be included in the same maximum clique. The independent sets are obtained from heuristic vertex-coloring, where each set constitutes a color class. The color classes are then used to prune branches of the maximum clique search tree.

Another relevant work related to classification using graph partitioning is transductive learning via spectral graph partitioning [5]. In [6] Vapnik introduced transductive Support Vector Machines (SVM). The transductive setting is different from the regular inductive setting since in this approach classification algorithm uses not only training patterns, but also test patterns and can potentially exploit structure in their distribution.

In [7] a graph partition algorithm is proposed. It uses the min-max clustering principle with a simple min-max function: the similarity between two subgraphs is minimized, while the similarity within each subgraph is maximized.

Another work addresses the solution of the supervised classification problem by reducing it to the solution of an optimization problem for partitioning of the graph on the minimal number of maximal cliques, [8]. This approach is similar to the one-versus-all SVM with a Gaussian radial basis function kernel, however unlike in the previous case no assumptions are made about statistical distributions of classes. The approach proposed in [8] differs from the integer programming formulation of the binary classification problem where the classification rule is a hyperplane which misclassifies the fewest number of patterns in the training set [9]. Initial results concerning the proposed approach have been presented in [10].

We can formulate the supervised classification problem as a *G-cut problem*. The feature space partitioning problem can be regarded as an  $n$ -dimensional cutting stock problem and is thus equivalent to making, say  $k_1$  guillotine cuts orthogonal to the  $x_1$  axis, then all  $k_1 + 1$  hyperparallelepipeds are cut into  $k_2$  parts by cuts orthogonal to the  $x_2$  axis, etc. Let us call such cuts “axes-driven-cuts”. Thus, if only axes-driven-cuts are allowed, the classification problem by parallel feature space partitioning could be stated as follows.

***G-cut Problem.*** *Divide an  $n$ -dimensional hyperparallelepiped into a minimal number of hyperparallelepipeds, so that each of them contains either patterns belonging to only one of the classes or it is the empty.*



Since the classes are separable according to their class label, the  $G$ -cut problem is solvable. This problem was first formulated and solved in [11] using parallel feature partitioning. The solution was obtained by partitioning the feature space into a minimal number of nonintersecting regions by solving an integer-valued optimization problem, which leads to the construction of minimal covering. The learning phase consists of geometrical construction of the decision regions for classes in  $n$ -dimensional feature space.

Let two training sets of patterns  $X$  and  $Y$  be given. We can consider them as points in the hypercube  $F \in R^n$ . Suppose that they are colored in blue and red, respectively.

During the learning phase the problem is to find for each group of points of the same color, for instance blue ones, a function  $f(\mathbf{x})$  for  $\mathbf{x} \in R^n$  such that the surface  $f(\mathbf{x}) = 0$  strictly separates the blue points from other points, i.e.  $f(\mathbf{x}) < 0$  for the blue ones and  $f(\mathbf{x}) > 0$  for the others.

If the two half spaces determined by the optimal hyperplane  $\mathbf{w} \cdot \mathbf{x} + b = 0$  are painted in red and blue, any new pattern is classified as red or blue, depending on the color of the corresponding half space. Thus, once the optimal hyperplane is found, the classification algorithm produces the output after  $n$  multiplications.

Nonlinear classifier looks for a function  $f$  and a constant  $b$  such that  $f(\mathbf{x}) < b$  for red points  $X$  and  $f(\mathbf{x}) > b$  for blue points  $Y$ . In the nonlinear case the notion of margin becomes complicated because the blue and red regions could not be connected. The problem can be illustrated by the following example.

**Example.** Let  $n = 1$  and the blue points in  $X$  are in the intervals  $[-6, -5] \cup [7, 12]$  and the red points in  $Y$  are in  $[-1, 3]$ . The classifier  $(x-1)^2 - 16 = 0$  paints  $[-3, 5]$  in red and its complement in blue. Let now  $\rho(x, y)$  be the distance between  $x$  and  $y$ . In this example the distance is  $|y - x|$ , but in general, the distance is depending on the norm chosen in  $R^n$ .

The problems with constructing of nonlinear classifiers  $f(\mathbf{x})$  are threefold:

- (i) the construction of  $f(\mathbf{x})$  should be computationally effective;
- (ii) the function has to be easily computable so that unknown patterns could be quickly classified;
- (iii) the function must yield large margins.

Next, we will consider the case when all patterns are points in  $R^n$ . The paper addresses the solution of the supervised classification problem by reducing it to heuristically solving good clique cover problem satisfying the nearest neighbor rule. First we apply a heuristic algorithm for partitioning a graph into a minimal number of cliques. Next cliques are merged using the nearest neighbor rule.

The rest of the paper is organized as follows. The class cover problem by colored boxes is discussed in Sect. 2. The supervised classification formulated as the minimum clique cover problem satisfying the nearest neighbor rule is described in Sect. 3. An algorithm for solving this problem is proposed in Sect. 4. Computational complexity of the proposed algorithm is discussed in Sect. 5 and classification rule is discussed in Sect. 6. Results of experiments are presented in Sect. 7. Finally, in Sect. 8 we draw some important conclusions.

## 2 Class Cover Problem by Colored Boxes

Recall that the patterns  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  are points in  $R^n$  and  $\mathbf{x} \in M$ , where  $M$  is the training set. In the sequel, the hyperparallelepiped  $P = \{X = (x_1, x_2, \dots, x_n), X \in I_1 \times I_2 \times \dots \times I_n\}$ , where  $I_i$  is a closed interval, will be referred to as a box. Suppose that the set  $K_c$  of patterns belonging to class  $c$  are painted in color  $c$ . For any compact  $S \subset R^n$ , let us denote by  $P(S)$  the smallest (in volume) box containing the set  $S$ , i.e.  $I_i = [l_i, u_i]$ , where  $l_i = \min x_i, \mathbf{x} \in S$  and  $u_i = \max x_i, \mathbf{x} \in S$ . A box  $P^c(*)$  is called painted in color  $c$ , if it contains at least one pattern  $\mathbf{x} \in M$  and all patterns in the box are of the same color  $c$ , i.e.  $P^c(*) \cap M \neq \emptyset$  and  $P^c(*) \cap M \subset K_c$ . Under these notations, we obtain the following Master Problem (MP):

*MP: Cover all points in  $M$  with a minimal number of painted boxes.*

Note that in the classification phase, a pattern  $\mathbf{x}$  is assigned to a class  $c$ , if  $\mathbf{x}$  falls in some  $P^c(*)$ . It is not necessary to require non-intersecting property for equally painted boxes. Suppose now that  $P(c) = \{P^c(S_1), P^c(S_2), \dots, P^c(S_{t_c})\}$  (minimal set of boxes of color  $c$ , covering all  $c$  colored points) is an optimal solution to the following problem:

*MP(c): Find the minimal cover of the points painted in color  $c$  by painted boxes.*

Then, one can easily prove that  $\cup P(c)$  (minimal cover) is an optimal solution to *MP*. Thus *MP* is decomposable in *MP(c)*,  $c = 1, 2, \dots, l$ . In [8] the *MP(c)* problem has been considered as a problem of partitioning the vertex set of a graph into a minimal number of maximal cliques.

In the next section we will show the relation of the *MP(c)* problem to the nearest neighbor rule.

## 3 Relation to the Nearest Neighbor Rule

A reasonable classification rule, known as a nearest neighbor rule, is to classify the pattern  $\mathbf{x}$  as red if  $\operatorname{argmin}_{\mathbf{y} \in X \cup Y} \rho(\mathbf{x}, \mathbf{y}) = \mathbf{y}^*$  and  $\mathbf{y}^*$  is red.

One could easily verify that any shift or scaling of the graphic in the example given in the Introduction  $(x - 1)^2 - 16 = 0$  will cause violation of the nearest neighbor rule for points falling in the margins  $(-5, -3)$  and  $(5, 7)$ .

In other words, a good classifier decomposes  $F$  into painted areas (in linear case they are only two) having the nearest neighbor property, i.e. for any point in red (blue) area the nearest neighbor rule classifies the recognized pattern as red (blue).

If box  $B = l_i \leq x_i \leq u_i$   $i = 1, \dots, n$  contains training patterns and  $\rho$  is the Manhattan distance, then for the pattern  $\mathbf{y}$  the distance is equal to  $\rho(\mathbf{y}, B) = \sum \max(0, l_i - y_i) + \max(0, y_i - u_i)$ . Now the idea of previously defined boxes becomes clear. We first approximate the above mentioned painted areas (not known in advance) by painted boxes (perfect candidates for Manhattan distance) and then classify patterns according to point-to-box distance rule.

Now the *MP(c)* problem can be formulated as an heuristic good clique cover problem satisfying the nearest neighbor rule.

## 4 A Clique Cover Algorithm

To introduce the algorithm we need to introduce additional notation. Consider again the master problem  $MP(c)$ . Let  $B = \{\mathbf{x} : l_i \leq x_i \leq u_i, i = 1, \dots, n\}$ . If  $u_i - l_i > 0, i = 1, \dots, n$  then we call the box  $B$  a **full dimensional box**. Suppose that two sets  $X$  and  $Y$  of training patterns (points in the hypercube  $F \in R^n$ ) are given and suppose that they are colored in blue and red, respectively.

We will call the box  $B$  **colored** iff it only contains points of the same color. A pair of points  $\mathbf{y} = (y_1, y_2, \dots, y_n)$  and  $\mathbf{z} = (z_1, z_2, \dots, z_n)$  generates  $B$  if  $l_i = \min\{y_i, z_i\}$  and  $u_i = \max\{y_i, z_i\}, i = 1, \dots, n$ .

**Problem A:** Find a coverage of  $X \cup Y$  with the minimal number of colored full dimensional boxes.

Define a graph  $G_X = (V, E), V = X, E = \{e = (v_i, v_j)\}$  and let  $e$  be a colored box generator. An edge  $e$  is colored green if it is a full dimensional box generator.

Let now  $e = (a, b)$  and  $f = (c, d)$  be green and let  $B_e$  and  $B_f$  are the corresponding full dimensional boxes. An operation  $e \oplus f$  is color preserving if the full dimensional box  $C, C = B_e \oplus B_f, l_i = \min\{a_i, b_i, c_i, d_i\}, u_i = \max\{a_i, b_i, c_i, d_i\}$  is colored. An edge  $e$  dominates  $f$  (say  $e > f$ ) if  $B_e \supset B_f$ .

Obviously, there is one-to-one correspondence between full dimensional boxes and the green edges. The dominance relation on the set of full dimensional boxes (say  $B_e > B_f$ ) could be easily established. When the full dimensional box  $C$  is colored then it dominates  $B_e$  and  $B_f$  and the appropriate application of  $\oplus$  operation allows generation of maximal colored cliques.

We call a clique colored if it contains green edges. The points contained in the full dimensional box  $C$  form the minimum clique cover, i.e., the vertex set (points in  $C$ ) is partitioned in cliques and the number of cliques is minimal.

Now we can reformulate the Problem A as follows.

**Problem A:** Cover the graph  $G_X$  with the minimum number of colored cliques.

The algorithm for solving Problem A is as follows.

- Step 1.** (Build the graph) Create the partial subgraph of  $G_X$  from the list **GE** of all green edges.
- Step 2.** (Clique enlargement) Create a graph  $GG_X = (V_{GG}, E_{GG})$ , where  $V_{GG} = \{v \in \mathbf{EGE}\}$  and  $E_{GG} = \{(e, f), B_e \oplus B_f\}$  is colored. Call **try-to-extend** (c).
- Step 3.** (Save the cliques (full dimensional boxes)) If **EGE** is the list of all extended boxes then discard from **GE** all  $e$  not included in **EGE**. Save the set **EGE**  $\cup$  **GE**. If all nodes are covered then stop else goto **Exceptions**.

**try-to-extend** (c): In all connected components of  $GG_X$  find  $c$ -clique cover (cliques of size less or equal to  $c$ ).

**Exceptions.** This function will be called if the set  $X$  is not coverable by the full dimensional boxes only. This case could be resolved by the algorithm above applied on the reduced

$X$  by covering it with lower dimensional boxes. Extreme instances when all nodes of  $G_X$  are singletons (nodes with degree one) will require rotation of the set  $X$  and are not discussed here.

**Remark:** singletons correspond to boxes of zero dimension and without rotation the box approach becomes the nearest neighbor approach.

## 5 Computational Complexity

Like many other methods, the optimal solution to the graph partitioning problem is  $NP$ -complete because of its combinatorial nature. While in both versions of the above-mentioned graph algorithm there is a call to a solver of a classical  $NP$ -complete problem, it is far from evident that the instances of  $MP(c)$  are not polynomially solvable. This is due to the fact that the vertices of the generated graphs are points in a metric space and clustering the points according to the Euclidian distance could result in forming cliques in the respective graphs.

We would like to point out that a new platform for solving the classification problem has been proposed, which in the exact case leads to solving an  $NP$ -complete problem. This can be avoided if approximate solution is sought.

To shed light on algorithm complexity, consider the following puzzle. Let paint an arbitrary subset of cells of a chessboard-like grid in blue and call blue piece a sequence of consecutive (horizontally or vertically) blue cells. The problem is to find the minimal number of blue pieces that cover all blue cells. If the length of the blue pieces is restricted by a constant  $c$  then so called absolute gap could be large. In integer programming this term is called a duality gap  $z^c - z^*$ . In this definition  $z^c$  is the optimal number of blue pieces of restricted length and  $z^*$  is the optimal number of blue pieces.

The lower bound of  $z^*$  which is equal to the minimal number of rows and columns which cover all blue cells can be found in a polynomial time. Algorithms for strip covering are considered in [12].

To come closer to the optimization problem in the graph  $GG_X$  let us define a rectangle consisting of blue cells only. If it is possible to find a good lower bound then this bound could be used to estimate the absolute gap. This estimate can be used for evaluation of acceptance of this heuristic solution.

To make the correspondence of each instance of such a puzzle with the classification problem in  $R^2$ , in the next step we will redefine pieces in an obvious way. To keep the polynomial complexity of the algorithm we sacrifice the optimality by using the threshold  $c$  as a parameter in try-to-extend procedure. Call now the speed-up  $s_{up} = |X|/|NB|$ , where  $NB$  is the cardinality of the clique cover. Since the above approach is the nearest neighbor in disguise, the bigger  $s_{up}$  is the faster classification procedure will become.

Step 1 finds a clique cover in  $O(|X|^3)$  time. To keep this complexity in practical use of the algorithm, one could adjust the threshold  $c$  to achieve a satisfactory  $s_{up}$ . Note that the main idea of the algorithm is to reduce the size of the clique cover problem on a graph with  $|X|$  nodes to much smaller size  $|GG_X|$ , which is decomposed into its connected components.

We would like to point out that the proposed new classifier is more general than the linear classifier. Note that considering blue and not blue points only doesn't diminish the applicability of the approach to more than two classes of patterns. In case of  $l$  classes for some integer  $l > 2$ , our classifier is applied sequentially for each class separately. The class membership is only used in the process of building  $G_c$ . This fact shows another advantage of the proposed algorithm.

## 6 Classification Rule

*Cliques-to-Painted Boxes.* Let  $S$  be any clique in the optimal solution of  $MP(c)$ . The box painted in color  $c$  that corresponds to this clique is defined by  $P(S) = \{\mathbf{x} = (x_1, x_2, \dots, x_n), \mathbf{x} \in I_1 \times I_2 \times \dots \times I_n\}$ , where  $I_i = [\min \bar{x}_i, \max \bar{x}_i]$ . The points  $\mathbf{x}$  correspond to the vertices in  $S$ . Geometrically, by converting cliques to boxes, one could obtain overlapping boxes of the same color. The union of such boxes is not a box, but in the classification phase the point being classified is trivially resolved as belonging to the union of boxes instead of a single box. If a pattern  $\mathbf{x}$  from the test dataset falls in a single colored box or in the union of boxes with the same color the element  $\mathbf{x}$  is assigned to the class that corresponds to this color.

If a pattern  $\mathbf{x}$  from the test dataset falls in an empty (uncolored) box then the pattern  $\mathbf{x}$  is not classified. Another possible classification rule is that the pattern  $\mathbf{x}$  can be assigned to a class with color that corresponds to the majority of adjacent colored boxes.

## 7 Experimental Results

In this section we compare the performance of our box algorithm and SVM classifier for synthetic data generated from 3-variate normal distributions and for real Monk's Problems data from UCI Machine Learning Repository.

### 7.1 Normal Attributes

The samples for a binary classification problem are generated for three cases and with 3-dimensional normal distributions with mean vectors and covariance matrices given in Table 1 below. where  $e = (1, 1, 1)^T$ . For each distribution 100 samples are generated and they are divided into 50 training samples and 50 testing samples. The simulation results are presented in Table 2 below.

**Table 1.** Parameter settings

Case	Covariance matrices	Mean vectors
1	$I \ I$	$0 \ 0.5e$
2	$I \ 2I$	$0 \ 0.6e$
3	$I \ 4I$	$0 \ 0.8e$

**Table 2.** Confusion matrices in percentage ratio for box algorithm and SVM classifier for normal data

	Box algorithm		SVM classifier	
	First normal distribution			
	Red points	Blue points	Red points	Blue points
Red points	<b>68.16</b>	31.84	67.10	32.90
Blue points	34.30	65.70	32.94	<b>67.06</b>
	Second normal distribution			
	Red points	Blue points	Red points	Blue points
Red points	72.84	27.16	<b>74.92</b>	25.08
Blue points	36.24	<b>63.76</b>	40.92	59.08
	Third normal distribution			
	Red points	Blue points	Red points	Blue points
Red points	<b>83.22</b>	16.78	83.12	16.88
Blue points	28.66	<b>71.34</b>	41.56	58.44

In Table 2 we use SVM with the standard Gaussian kernel. It can be noticed that in most cases the box algorithm outperforms SVM classifier in terms of true positive and true negative rates. For example, its advantage is 13% for the true negative rate for blue points from the third normal distribution.

## 7.2 Nominal Attributes

In this section we present experimental results on three Monk's Database problems from UCI Machine Learning Repository. Each problem consists of training and testing data samples with the same 6 nominal attributes. Data sizes are as follows: Monk1 - 124, Monk2 - 169, Monk3 - 122 (train) and Monk1 - 432, Monk2 - 432, Monk3 - 432 (test), respectively. In Table 3 we used SVM classifier with the standard Gaussian kernel. A 10-fold cross validation yields error 0.33 for Monk1 and Monk2.

It can be noticed that in most cases the box algorithm clearly outperforms SVM classifier in terms of true positive and true negative rates. For example, its advantage for Monk1 is 33% and 15% for the true positive and true negative rates, respectively.

It can be observed in Table 4 that the box algorithm achieves better accuracy than SVM classifier for normal distributions and Monks and furthermore it achieves better sensitivity for almost all normal distributions and Monks.

One can notice in Table 5 that in most cases the box algorithm achieves better or the same specificity and precision as SVM classifier for normal distributions and Monks.

Consequently, it can be seen from the experimental results presented in this section that the box algorithm is superior to SVM in almost all cases.

**Table 3.** Confusion matrices in percentage ratio for box algorithm and SVM classifier for Monks data

	Box algorithm		SVM classifier	
	Monk1			
	Red points	Blue points	Red points	Blue points
Red points	<b>100</b>	0	66.67	33.33
Blue points	20.37	<b>79.63</b>	35.19	64.81
	Monk2			
	Red points	Blue points	Red points	Blue points
Red points	<b>55.86</b>	44.14	47.93	52.07
Blue points	36.62	<b>63.38</b>	41.55	58.45
	Monk3			
	Red points	Blue points	Red points	Blue points
Red points	88.24	11.76	<b>89.71</b>	10.29
Blue points	21.05	<b>78.95</b>	25.88	74.12

**Table 4.** Accuracy and sensitivity of SVM classifier and the box algorithm for Monks and normal data

	Normal distributions			Monks		
	Accuracy					
	1	2	3	1	2	3
SVM classifier	0.67	0.67	0.71	0.66	0.53	0.82
Box algorithm	0.67	<b>0.68</b>	<b>0.77</b>	<b>0.90</b>	<b>0.60</b>	<b>0.84</b>
	Sensitivity					
	1	2	3	1	2	3
SVM classifier	<b>0.67</b>	0.59	0.58	0.65	0.58	0.79
Box algorithm	0.66	<b>0.64</b>	<b>0.71</b>	<b>0.80</b>	<b>0.63</b>	0.79

**Table 5.** Specificity and precision of SVM classifier and the box algorithm for Monks and normal data

	Normal distributions			Monks		
	Specificity					
	1	2	3	1	2	3
SVM classifier	0.67	<b>0.75</b>	0.83	0.67	0.48	<b>0.90</b>
Box algorithm	<b>0.68</b>	0.73	0.83	<b>1</b>	<b>0.56</b>	0.88
	Precision					
	1	2	3	1	2	3
SVM classifier	0.67	0.70	0.78	0.66	0.53	<b>0.88</b>
Box algorithm	0.67	0.70	<b>0.81</b>	<b>1</b>	<b>0.59</b>	0.87

## 8 Conclusions

We introduced a new geometrical approach for solving the supervised classification problem. We applied graph optimization approach using the well-known problem of partitioning the graph into a minimum number of cliques which were subsequently merged using the nearest neighbor rule. Equivalently, the supervised classification problem is solved by means of a heuristic good clique cover problem satisfying the nearest neighbor rule. The main advantage of the new approach which optimally utilizes the geometrical structure of the training set is decomposition of the  $l$ -class into  $l$  single-class optimization problems. The computational complexity of the proposed algorithm, the computational procedure, and the classification rule are discussed. One can see that the box algorithm performs better than SVM in almost all cases. A geometrical interpretation of the solution and simulation examples are also given.

As a future work we are planning to compare the computational efficiency of the proposed algorithm with the classical classification techniques such as decision trees, ensembles of trees, and random forest.

## References

1. Yang, G., Tang, Z., Zhang, Z., Zhu, Y.: A Flexible annealing chaotic neural network to maximum clique problem. *Int. J. Neural Syst.* **17**(3), 183–192 (2007)
2. Wang, R.L., Tang, Z., Cao, Q.P.: An efficient approximation algorithm for finding a maximum clique using hopfield network learning. *Neural Comput.* **15**(7), 1605–1619 (2003)
3. Pelillo, M., Torsello, A.: Payoff-Monotonic game dynamics and the maximum clique problem. *Neural Comput.* **18**(5) (2006)
4. Kumlander, D.: Problems of optimization: an exact algorithm for finding a maximum clique optimized for dense graphs. In: *Proceedings of the Estonian Academy of Sciences, Physics, Mathematics*, vol. 54, no. 2, pp. 79–86 (2005)
5. Joachims, T.: Transductive learning via spectral graph partitioning. In: *Proceedings of Twentieth International Conference on Machine Learning*, pp. 290–297, Washington DC (2003)
6. Vapnik, V.: *Statistical Learning Theory*. Wiley, Hoboken (1998)
7. Ding, C.H.Q., He, X., Zha, H., Gu, M., Simon, H.D.: A min-max cut algorithm for graph partitioning and data clustering. In: *Proceedings of International Conference on Data Mining*, pp. 107–114 (2001)
8. Valev, V., Yanev, N.: Classification using graph partitioning. In: *Proceedings of the 21st International Conference on Pattern Recognition*, pp. 1261–1264 (2012)
9. Yanev, N., Balev, S.: A combinatorial approach to the classification problem. *Eur. J. Oper. Res.* **115**(2), 339–350 (1999)
10. Valev, V., Yanev, N., Krzyżak, A.: A new geometrical approach for solving the supervised pattern recognition problem. In: *Proceedings of the 23rd International Conference on Pattern Recognition*, pp. 1648–1652 (2016)
11. Valev, V.: Supervised pattern recognition by parallel feature partitioning. *Pattern Recogn.* **37**(3), 463–467 (2004)
12. Ghasemi, T., Ghasemalizadeh, H., Razzazi, M.: An algorithmic framework for solving geometric covering problems - with applications. *Int. J. Found. Comput. Sci.* **25**(5), 623–639 (2014)





# Deep Homography Estimation with Pairwise Invertibility Constraint

Xiang Wang<sup>1</sup>, Chen Wang<sup>1</sup>, Xiao Bai<sup>1(✉)</sup>, Yun Liu<sup>2</sup>, and Jun Zhou<sup>3</sup>

<sup>1</sup> School of Computer Science and Engineering, Beihang University, Beijing, China  
stanleywang.0.1@outlook.com, {wangchenbuaa, baixiao}@buaa.edu.cn

<sup>2</sup> School of Automation Science and Electrical Engineering, Beihang University,  
Beijing, China

<sup>3</sup> School of Information and Communication Technology, Griffith University,  
Nathan, Australia

**Abstract.** Recent works have shown that deep learning methods can improve the performance of the homography estimation due to the better features extracted by convolutional networks. Nevertheless, these works are supervised and rely too much on the labeled training dataset as they aim to make the homography be estimated as close to the ground truth as possible, which may cause overfitting. In this paper, we propose a Siamese network with pairwise invertibility constraint for supervised homography estimation. We utilize spatial pyramid pooling modules to improve the quality of extracted features in each image by exploiting context information. Discovering the fact that there is a pair of homographies from a given image pair which are inverse matrices, we propose the invertibility constraint to avoid overfitting. To employ the constraint, we adopt the matrix representation of the homography rather than the commonly used 4-point parameterization in other methods. Experiments on the synthetic dataset generated from MSCOCO dataset show that our proposed method outperforms several state-of-the-art approaches.

**Keywords:** Homography estimation · Supervised deep learning  
Invertibility constraint · Spatial pyramid pooling

## 1 Introduction

Homography estimation is one of fundamental geometric problems and is widely applied to many computer vision and robotics tasks such as camera calibration, image registration, camera pose estimation and visual SLAM [1–4]. A 2D homography relates two images capturing the same planar surface in 3D space from different perspectives by mapping one image to the other. Thus the homography indicates the camera pose transformation which is a key factors in many tasks. For example, in visual SLAM methods such as ORB-SLAM [5], homography estimation is one of the options for camera motion initialization, especially in some degenerate configurations, such as planar or approximately planar scenes,

and rotation-only camera motions. To boost a visual SLAM system successfully, a fast, accurate and robust homography estimation approach is demanded.

Traditional homography estimation method can be categorized as feature-based methods and direct methods. Feature-based methods first detect keypoints in each image and generate reliable feature descriptors such as SIFT [6] and ORB [7] features. Then feature correspondences between keypoint sets in two images are established by feature matching. The homography between these two images is estimated by RANSAC [8] which generates multiple options and choose the one with the minimum mapping error. Feature-based methods are the mainstream methods because of better accuracy. However, feature-based methods rely too much on the features, both in effectiveness and in efficiency. When keypoints cannot be successfully extracted because of lack of texture, or wrong feature correspondences exist due to occlusions, repetitive textures or illumination changes, the correctness of estimated homography can be significantly degraded. Moreover, to maintain the distinctiveness and invariance of features, the computation of man-made descriptors can be slow, leading to efforts of designing time-saving descriptors while having worse performance.

Direct methods, such as Lucas-Kanade algorithm [9], use all pixels rather than a few keypoints to establish correspondences between two images. The standard pipeline is a pixel-to-pixel matching, initialized by warping one image to another using a homography guess and followed by an iterative photometric error minimization with an error metric such as the sum of squared differences (SSD) and an optimization technique such as Gauss-Newton method or gradient descent [10]. By utilizing all pixels over the images, the accuracy and robustness of direct methods can be comparable to feature-based ones, while coming with more computational cost and thus being slower.

Deep Convolutional Neural Network (CNN) methods have seen rapid development and successful applications in many geometric computer vision problem such as optical flow estimation [11], stereo matching [12], camera localization [13], monocular depth estimation [14] and visual odometry [15]. CNN can be regarded as a powerful image feature extractor which extracts more distinctive features than direct methods and still maintains information of the whole image rather than only preserving local features in feature-based methods, thus shows promising potential for improving the performance of homography estimation both in accuracy and in robustness. DeTone et al. [16] firstly utilize a VGG-like CNN to tackle the homography estimation problem. The HomographyNet can be decomposed to two parts: a feature extractor and a regressor/classifier to get the final estimation. Both parts can be learned given the supervised ground truth labels of the homography generated by manually warping an given image. Then, the learned model starts with stacking two image patches together as input, and processes them through the network to get a 4-point homography estimation. Nowruzi et al. [17] use a hierarchical CNN architecture to reduce the error bounds of the homography estimation. The model starts with a Siamese architecture to extract features of two image patches independently and merges them later to get a rough homography estimate. To reduce the estimation error, an iterative

scheme is applied, leading to a hierarchical architecture of the network and an iteratively updated homography estimate. Recently, Nguyen et al. [18] propose an unsupervised method for homography estimation by minimizing a pixel-wise intensity error metric between the target image and the warped one using the estimated homography. Similar ideas can be seen in conventional direct SLAM methods [19] and the unsupervised deep learning method for monocular depth and camera pose estimation [20]. However, without labeled data as ground truth, the estimation is not as accurate as that of supervised learning method. Besides, the labeled data can be generated relatively easily, which reduces the significance of unsupervised learning of this task to some extent.

In this paper, we propose a supervised method to improve the accuracy of homography estimation from a given image pairs using convolutional neural networks. By employing a spatial pyramid pooling module inspired by the work of stereo matching [21], feature extracting performance of the convolutional parts can be improved due to exploiting context information of the image. Moreover, we make a full use of an image pair in the training set by giving bidirectional homography estimation. That will produce two homographies which are inverse matrices. We explicitly combine this invertibility constraint into the loss function to improve the performance. We argue that the common 4-point homography parameterization in other deep learning method is not suitable for the proposed invertibility constraint, and we choose the classical matrix parameterization instead. We show that the proposed network and the loss function improve the accuracy of the results.

Our main contributions are as follows:

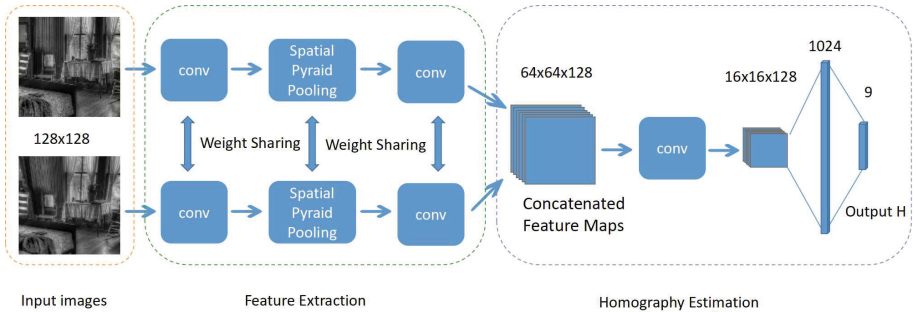
- We propose a modified end-to-end learning framework for deep homography estimation by using a Siamese architecture and spatial pyramid pooling modules. It is the first time that spatial pyramid pooling is integrated to solve the homography estimation problem.
- We estimate two homographies from one image pair and make use of the inherent invertibility of them into the loss function to avoid overfitting.
- We perform experiments and show that our methods achieve better accuracy of the results and the employment of the invertibility contributes to the results.

## 2 The Proposed Model

In this section, we present in detail the network architecture and the loss function we propose. The aim of our network is to estimate the homography between two given images in an end-to-end manner. The image pair is firstly sent to the Siamese architecture for feature extraction independently. These features are then stacked and sent to another convolutional part to get pairwise relations. The final fully connected layers are employed for the final estimated homography. Details are given in the following subsections.

## 2.1 Network Architecture

The network takes two normalized image patches in size of  $128 \times 128$  pixels as input. We adopt a Siamese network architecture, which uses 4 convolutional layers as the first feature extractor part to treat two patches separately while sharing the weights of these two streams to achieve the same feature extraction result, and then uses another 4 convolutional layers as the second feature extractor part after stacking two feature maps together to explore the relation between these two images. Each convolutional layer consists of the basic  $3 \times 3$  residual convolutional block with Batch Normalization and ReLUs, with a max pooling layer after the fourth and the sixth convolutional layers. Among these layers, a spatial pyramid pooling module is inserted after the second convolutional layer, in order to capture different size of objects, especially in the case that there is a belonging relationship between an object and its sub-regions. The pyramid module incorporates the hierarchical context relationship to the extracted features rather than only have features from pixel intensities. In our work, we adopt the similar spatial pyramid pooling design pattern as [21] which tackles the stereo matching problem for depth estimation. The pyramid has four fixed-size average pooling blocks:  $64 \times 64$ ,  $32 \times 32$ ,  $16 \times 16$ , and  $8 \times 8$ , followed by  $1 \times 1$  convolution and upsampling. After concatenating two feature pyramid channel-wise, the tensor are sent to the second part to extract correlations between these two image patches, similar to the traditional feature matching procedure. Then two fully-connected layers are followed with a dimensionality of 1024 and 9 to get a real-valued vectorized homography estimate as the output. To avoid overfitting, a dropout scheme with a drop probability of 0.5 is employed after the last convolutional layer and the first fully-connected layer. The detail of the network architecture is illustrated in Fig. 1.



**Fig. 1.** Network architecture for our proposed method. The network processes an image pair twice with the order of the pair changing to get two estimated vectorized homographies  $\mathbf{h}_{12}$ ,  $\mathbf{h}_{21}$ . Then we can utilize the invertibility constraint to this pair of homographies after normalization and matrixing.

## 2.2 Invertibility Constraint of Homography

To enhance the performance of the homography estimation, a possible way is to independently estimate two homographies related to the given image pair. That is, given an image pair  $I^A$  and  $I^B$ , a homography  $\mathbf{H}^{BA}$  can be checked by warping  $I^A$  to a synthetic image that is close to the target  $I^B$ , and also  $I^B$  can be warped to  $I^A$  given the homography  $\mathbf{H}^{AB}$ . Both homography results are related to the same estimation scheme and the same input, except for the change of the image pair's order. In practical applications, both orders of the input image pair is valid. Therefore, by utilizing one image pair twice, the training test is doubled. With the promoted accuracy on the training dataset, there is potential for overfitting on the training set and bad generalization to new image pairs. Particularly, we are concerned that  $\mathbf{H}^{BA}$  and  $\mathbf{H}^{AB}$  may tend to be more correlated to the image information and the inherent relation between the homography pair is neglected. Note that  $\mathbf{H}^{BA}$  and  $\mathbf{H}^{AB}$  are inverse matrices, i.e.,  $\mathbf{H}^{BA}\mathbf{H}^{AB} = I$ , the invertibility constraint can be added to the loss function which encourages the network to produce an estimation that satisfies the complete bidirectional warping characteristic and thus avoids overfitting due to unidirectional transform for one image pair.

## 2.3 Parameterization of the Homography

Most deep learning homography estimation works use a 4-point homography parameterization based on the locations of the image patch corners [16–18]. The parameterization is derived from the image warping procedure. To obtain the warped target image, we need to know the pixel location  $(u, v)$  to be mapped in the target image and the corresponding pixel location  $(u', v')$  in the source image which have the desired pixel intensity. Then, the homography mapping is established up to scale. Given 4 pairs of selected image patch corners, the following equations can be solved using the normalized Direct Linear Transform (DLT) algorithm [22].

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{pmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} \sim \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} \quad (1)$$

Noticing that the homography has only 8 degrees of freedom, the matrix representation is over parameterized. The 4-point homography representation denote the homography as the pixel coordinate offsets  $(\Delta u, \Delta v) = (u - u', v - v')$  of 4 pairs of selected image patch corners. Actually, by fixing the pixel coordinates in the source frame, this representation is identical to the pixel coordinates in the target frame, and can be uniquely transformed to the conventional matrix representation. However, the values of the coordinate offsets depend on the coordinates in the source frame, which may cause an inconsistent homography estimate to other pixels inside the image patch. More importantly, the matrix representation is more suitable for our proposed invertibility constraint. The pair of computed pixel coordinate offsets, the 4-point homographies, are

desired to be opposite to form the additional constraint as the offsets in the image pair should indicate the same line segment in the scene. Nevertheless, that assumption fails as the viewpoints of the images have changed. Therefore, we adopt the conventional matrix representation rather than the 4-point parameterization.

## 2.4 Loss Function

Combining the invertibility loss with the original loss between the ground truth and the estimate of the homography, we can define the loss function as

$$loss = \frac{1}{2} \left\| \frac{\mathbf{h}_{12}}{h_{12}^{(9)}} - \mathbf{h}_{12}^* \right\|_2 + \frac{1}{2} \left\| \frac{\mathbf{h}_{21}}{h_{21}^{(9)}} - \mathbf{h}_{21}^* \right\|_2 + \frac{\lambda}{2} \|\mathbf{H}_{12}\mathbf{H}_{21} - \mathbf{I}\|_F. \quad (2)$$

where  $\mathbf{h}_{12}$  is the 9-dimensional output of the network which indicates the vectorized homography estimate from image 2 to image 1, and a similar notation  $\mathbf{h}_{21}$  is the vectorized homography from image 1 to 2.  $h_{12}^{(9)}$  is the ninth dimension of the output vector and the output is divided by it for normalization.  $\mathbf{H}_{12}$  denoted the estimated matrix transformed from the normalized vector.  $\mathbf{h}_{12}^*$  denotes the ground truth of the normalized homography vector that is given during the generation of the training dataset.  $\mathbf{I}$  is the identity matrix. And  $\lambda$  is the weighting parameter that balances the impact of the error terms and the invertibility constraint. We choose  $L_2$  loss function for the first two error terms and the Frobenius norm for the last one to keep the same loss metric among them.

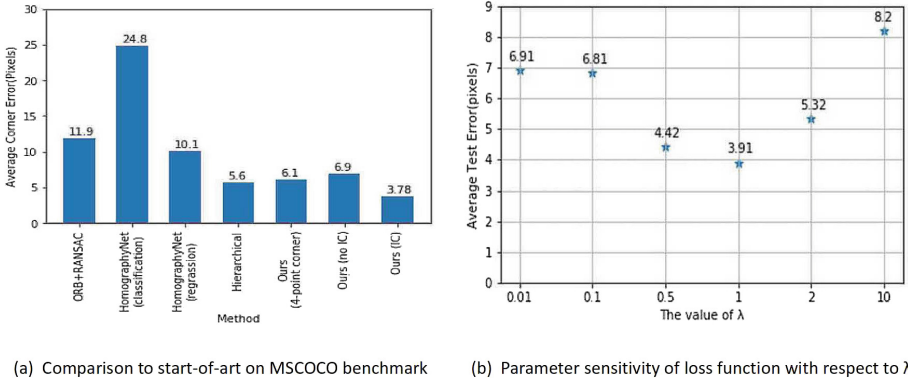
## 3 Experiments

In this section, we evaluate the performance of our proposed method on the synthetic dataset generated from the MSCOCO dataset. We compare our method to both the traditional method and supervised deep learning methods in terms of the corner error. Further analysis and experiments are shown for the influence of different parameterizations and the choice of the balancing parameter between error terms and the invertibility constraint. We also visualize the results of our method.

### 3.1 Dataset Description

We evaluate our method on the dataset constructed based on the commonly used Microsoft Common Objects in Context (MSCOCO) 2014 dataset [23] as in [16]. The images in the dataset are converted to gray-scale and resized to a resolution of  $320 \times 240$ . We produce 5 patches from the given image by choosing random squares of a  $128 \times 128$  size within the image. To acquire the warped patches, we perform a perturbation on the patch corner points within the range of 32 pixels to determine which part the obtained image patches contain.

(The perturbed corner positions should be still within the image.) The corresponding homography can be derived as the ground truth from these 4 pairs of corner positions with the OpenCV library. By applying the homography to the given patches, the warped patches can be generated directly. Thus, we can get both the image patch pairs and the homography ground truth in the training and test dataset.



(a) Comparison to start-of-art on MSCOCO benchmark

(b) Parameter sensitivity of loss function with respect to  $\lambda$ 

**Fig. 2.** (a) The accuracy comparison of our proposed method to the state-of-the-art in terms of the Average Corner Error metric. The baselines are ORB+RANSAC, HomographyNet and Hierarchical Network. We also test our models when no invertibility loss is appended to the loss function (no IC) and when utilizing the common 4-point parameterization (4-point corner) without the invertibility constraint. The results show that all deep learning methods achieve better accuracy than the traditional ORB+RANSAC method except for HomographyNet (classification) which treats the homography estimation as a classification problem rather than a regression problem. Our method with the invertibility constraint (IC) and the matrix representation shows the best performance among all the methods. (b) The sensitivity test of the balancing parameter  $\lambda$  in the loss function. The optimum of  $\lambda$  lies around 1, and 0.9 is a more exact result after further experiments.

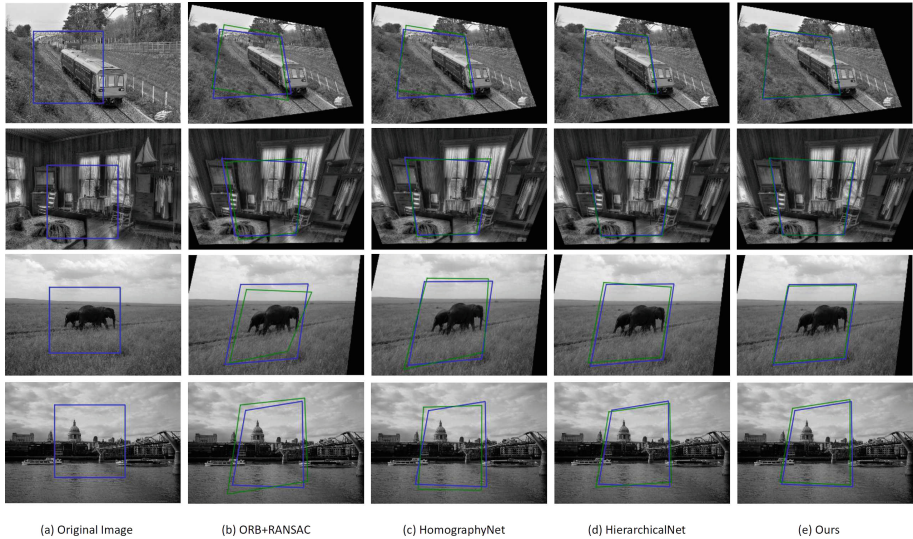
### 3.2 Experiment Implementation

We implement the proposed network using the publicly available PyTorch framework for all experiments. The model parameters are initialized using a uniform distribution and then optimized with Adam optimizer. The model is trained for 90,000 total iterations on a single Nvidia Titan X GPU with 64 images per mini-batch. We use a base learning rate of 0.005 and decrease it by a factor of 10 after every 30,000 iterations.

### 3.3 Experiment Results and Comparison

In this experiment, we compare our model to the following traditional or deep learning methods as the baselines. The first baseline is a traditional approach





**Fig. 3.** Visualization of the test samples. The quadrangles represent the warped image patches from the leftmost column of images, among which the blue ones are related to the homography ground truth and the green ones are related to the estimated homographies. Significantly all deep learning methods perform better than the traditional ORB+RANSAC scheme. And our proposed method achieves the best performance. (Color figure online)

based on feature matching with ORB descriptors followed by a robust RANSAC homography estimation scheme. The deep learning baselines are the HomographyNet proposed by [16] and the hierarchical network presented in [17], both of which are supervised methods like the method we propose.

The result are shown in Fig. 2(a). We use the Mean Average Corner Error as the error metric for each approach. To gain that, the  $L_2$ -distance between the ground truth and the estimate of the corner position is firstly computed, and then the averaged error is computed over the four corners of the given image. The final mean is calculated over the entire test set. We found that our full implementation performs the best compared to other baselines, especially to the hierarchical homography network [17] which has a similar architecture to our network. That proves the effectiveness of our invertibility constraint. And all the regression networks for homography estimation outperform the traditional ORB+RANSAC method due to better feature matching results. The visualized results of homography estimation are illustrated in Fig. 3.

To investigate the impact of invertibility constraint, we also evaluate the performance of our network without it. In Fig. 2(a) we find that without the invertibility constraint, the accuracy is lower than the hierarchical homography network. Although the spatial pyramid pooling module may take effect, it doesn't lower the error bound of homography, which can be achieved by the hierarchical architecture. That will lead to higher potential for inaccurate estimates.



Moreover, different parameterizations can also influence the performance of the network. We conduct an additional experiment using the 4-point representation without the invertibility constraint. We find that under the same network architecture and loss function (no invertibility constraint), 4-point parameterization indeed outperforms the matrix representation, consistent with the conclusion in [24]. Thus the invertibility constraint can improve the performance with the matrix representation over the 4-point parameterization.

### 3.4 Evaluation of the Balancing Parameter $\lambda$

Another question is how to balance these two parts of the loss, the error terms and the invertibility loss. In other words, which value should we choose for the balancing parameter  $\lambda$ ? Figure 2(b) shows some tests on the accuracy of our method when changing the value of  $\lambda$ . Clearly, there is an optimum for  $\lambda$  around 1. By tuning  $\lambda$  between 0.8 and 1.2 with a step of 0.1, the best value is identified as  $\lambda = 0.9$ . As the value gets smaller, the invertibility constraint has less influence on the final estimation and the method tend to be similar like previous methods which may cause overfitting to the training dataset. On the other hand, when  $\lambda$  becomes larger, the training set will take less effect and the final homography matrix estimation will be close to the identity  $\mathbf{I}$  which definitely fits to the invertibility constraint but is not desired.

## 4 Conclusion

In this paper, we have presented a novel end-to-end model for homography estimation using a convolution neural network. We argue that reusing the given image pair can double the training set and give potential for more constraint of homography estimation. Besides the common error term between the ground truth and estimates of the homography, we add an extra invertibility constraint loss to the training loss function in order to maintain the inherent property of the homography and avoid overfitting to the training set. To apply this constraint, the 4-point parameterization of homography commonly used in other deep learning methods cannot be accepted and we choose to utilize the conventional matrix homography representation. Experiments on the synthetic dataset generated from MSCOCO dataset show a promotion to the accuracy of homography estimation compared to the state-of-the-art deep learning approaches. Although the matrix representation itself cannot give a better performance to the task compared to the 4-point parameterization, the accuracy can be improved when accompanied by the additional invertibility constraint.

**Acknowledgement.** This work was supported by the National Natural Science Foundation of China project no. 61772057, in part by Beijing Natural Science Foundation project no. 4162037, and the support funding from State Key Lab. of Software Development Environment.

## References

1. Song, Y.Z., Xiao, B., Hall, P., et al.: In search of perceptually salient groupings. *IEEE Trans. Image Process.* **20**(4), 935–947 (2011)
2. Liu, S., Bai, X.: Discriminative features for image classification and retrieval. *Pattern Recognit. Lett.* **33**(6), 744–751 (2012)
3. Bai, X., Ren, P., Zhang, H., et al.: An incremental structured part model for object recognition. *Neurocomputing* **154**, 189–199 (2015)
4. Liang, J., Zhou, J., Tong, L., et al.: Material based salient object detection from hyperspectral images. *Pattern Recognit.* **76**, 476–490 (2018)
5. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Trans. Robot.* **31**(5), 1147–1163 (2015)
6. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **60**(2), 91–110 (2004)
7. Rublee, E., Rabaud, V., Konolige, K., et al.: ORB: an efficient alternative to SIFT or SURF. In: 2011 IEEE International Conference on Computer Vision, ICCV, pp. 2564–2571. IEEE (2011)
8. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In: *Readings in Computer Vision*, pp. 726–740 (1987)
9. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, vol. 2, pp. 674–679. Morgan Kaufmann Publishers Inc. (1981)
10. Baker, S., Matthews, I.: Lucas-Kanade 20 years on: a unifying framework. *Int. J. Comput. Vis.* **56**(3), 221–255 (2004)
11. Dosovitskiy, A., Fischer, P., Ilg, E., et al.: FlowNet: learning optical flow with convolutional networks. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2758–2766 (2015)
12. Zbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.* **17**(1–32), 2 (2016)
13. Kendall, A., Grimes, M., Cipolla, R.: PoseNet: a convolutional network for real-time 6-DOF camera relocalization. In: 2015 IEEE International Conference on Computer Vision, ICCV, pp. 2938–2946. IEEE (2015)
14. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: *CVPR*, vol. 2, no. 6, p. 7 (2017)
15. Wang, S., Clark, R., Wen, H., et al.: DeepVO: towards end-to-end visual odometry with deep recurrent convolutional neural networks. In: 2017 IEEE International Conference on Robotics and Automation, ICRA, pp. 2043–2050. IEEE (2017)
16. DeTone, D., Malisiewicz, T., Rabinovich, A.: Deep image homography estimation. *arXiv preprint [arXiv:1606.03798](https://arxiv.org/abs/1606.03798)* (2016)
17. Japkowicz, N., Nowruzki, F.E., Laganieri, R.: Homography estimation from image pairs with hierarchical convolutional networks. In: 2017 IEEE International Conference on Computer Vision Workshop, ICCVW, pp. 904–911. IEEE (2017)
18. Nguyen, T., Chen, S.W., Skandan, S., et al.: Unsupervised deep homography: a fast and robust homography estimation model. *IEEE Robot. Autom. Lett.* **3**, 2346–2353 (2018)
19. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: large-scale direct monocular SLAM. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) *ECCV 2014*. LNCS, vol. 8690, pp. 834–849. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-10605-2\\_54](https://doi.org/10.1007/978-3-319-10605-2_54)

20. Zhou, T., Brown, M., Snavely, N., et al.: Unsupervised learning of depth and ego-motion from video. In: CVPR, vol. 2, no. 6, p. 7 (2017)
21. Chang, J.R., Chen, Y.S.: Pyramid stereo matching network. arXiv preprint [arXiv:1803.08669](https://arxiv.org/abs/1803.08669) (2018)
22. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2003)
23. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-10602-1\\_48](https://doi.org/10.1007/978-3-319-10602-1_48)
24. Baker, S., Datta, A., Kanade, T.: Parameterizing homographies. Technical report CMU-RI-TR-06-11 (2006)

# **Spatio-temporal Pattern Recognition and Shape Analysis**



# Graph Time Series Analysis Using Transfer Entropy

Ibrahim Caglar<sup>(✉)</sup> and Edwin R. Hancock

Computer Vision and Pattern Recognition, Department of Computer Science,  
University of York, York YO10 5DD, UK  
ic656@york.ac.uk

**Abstract.** In this paper, we explore how Schreiber's transfer entropy can be used to develop a new entropic characterisation of graphs derived from time series data. We use the transfer entropy to weight the edges of a graph where the nodes represent time series data and the edges represent the degree of commonality of pairs of time series. The result is a weighted graph which captures the information transfer between nodes over specific time intervals. From the weighted normalised Laplacian we characterise the network at each time interval using the von Neumann entropy computed from the normalised Laplacian spectrum, and study how this entropic characterisation evolves with time, and can be used to capture temporal changes and anomalies in network structure. We apply the method to stock-market data, which represent time series of closing stock prices on the New York stock exchange and NASDAQ markets. This data is augmented with information concerning the industrial or commercial sector to which the stock belong. We use our method not only to analyse overall market behaviour, but also inter-sector and intra-sector trends.

## 1 Introduction

Recent work has shown that the entropic analysis of graph-time series, can lead to powerful tools for analysing their salient structure, distinct evolutionary epochs and the identification of anomalous events [18]. Graph entropy captures the structure of networks at a complexity level. For instance, highly random structures are associated with high entropy while non-random structures associated with low entropy. Moreover, if a principled measure of graph entropy is to hand then information theoretic measures such as the Kullback-Leibler and Jensen-Shannon divergences can be used to measure the similarity of different graphs and can lead to the definition of information theoretic graph kernels that can be used to embed graph time series into low-dimensional vector spaces [2, 3, 21]. Moreover, they allow statistical models of the time evolution of graphs to be learned. As a concrete example, Ye et al. have shown how to compute an approximation of the von Neumann entropy of a graph, using simple degree statistics [18]. Here the entropy associated with an edge in a graph depends on the reciprocal of the product of the node-degrees defining the edge.

One domain where the analysis of graph or network time series has proved particularly useful is the analysis of financial markets. Here the nodes represent different stock or trading entities, and edges indicate the similarity of trading patterns for a different stock. There are several ways to establish similarity over time. The simplest of these is to compute the correlation of time series of trading prices and to create an edge if the correlation exceeds a threshold value [19]. Alternatives include the use of Granger Causality [7] and most recently transfer entropy [15]. In fact, Granger causality was originally introduced in the financial domain and has recently found application in the brain-imaging domain where it has been used to establish network representations of brain activation patterns in fMRI data [17].

In this paper, we turn our attention to transfer entropy. The characterisation adopted by Ye et al. [20] and Bai et al. [2] in their work on times-series and kernel-based analysis of graphs, utilities von Neumann entropy to characterize the structure of the networks and time-series correlation to construct the edges of the network. Unfortunately, when posed in this way there is no information theoretic characterisation for the evidential support for the edges of the network. The aim of this paper is to fill this gap in the literature by developing a new characterisation of network entropy in which the edges are weighted to reflect their associated transfer entropy or information flow between nodes.

This leads us to a novel representation of network evolution with time. At each time epoch we construct a weighted graph in which the edge weights are computed from transfer entropies between pairs of nodes. This is an instantaneous time-snap of the pattern of information flow between nodes. We analyse time series by observing how this network structure evolves with time. We apply the method to financial market data. The newly constructed dataset contains 431 companies in 8 different commercial or industrial sectors from the NYSE and NASDAQ markets. There are about 50 stocks in each of 8 different sectors. These stock have the largest market capitalization in their respective sectors. The period covered by the data ends in December 2016 and covers about 20 years, and so the dataset covers 5500 trading days from January 1995. Several economic and market crises are covered by the data, including global financial crisis and European debt crisis. We use this data to analyse both the global structure of the trading network and details its sub-sector structure with time. This includes an analysis of how the inter-sector and intra-sector transfer entropy varies with time, and in particular how they change during the market crises listed above.

The outline of this paper is as follows. In Sect. 2 we introduce the basic definitions of transfer entropy and show how it can be used to characterise an edge in a graph. Section 3 details our graph-based representation drawing on transfer entropy. Section 4 provides experimental results. Section 5 offers some conclusions and directions for future research.

## 2 Edge Transfer Entropy from Times Series

### 2.1 Basic Definitions

To compute transfer entropy, we first require some basic concepts from information theory. Consider the random variable  $X$ , following a probability distribution  $p(x)$ , where  $x$  is particular values of  $X$ . The Shannon Entropy [16] of the distribution  $p(x)$  is defined as

$$H(X) = - \sum_x p(x) \log_2 p(x)$$

The base of the logarithm determines the units used for measuring information, and in base 2 the results are given bits [12] if base the is natural the results are given in nits [6].

The joint entropy of the random variables  $X$  and  $Y$  is defined as [1]

$$H(X, Y) = - \sum_x \sum_y p(x, y) \log_2 p(x, y)$$

and the conditional entropy of  $X$  given  $Y$  [1] is

$$H(X|Y) = - \sum_x \sum_y p(x, y) \log_2 p(x|y)$$

The mutual information of two random variables  $X$  and  $Y$  is  $I(X, Y) = H(X) + H(Y) - H(X, Y)$  or equivalently  $I(X, Y) = H(X) - H(X|Y)$  or  $I(X, Y) = H(Y) - H(Y|X)$  where  $H(X)$ ,  $H(Y)$  are the Shannon entropies and  $H(X, Y)$  the is joint entropy. Since the mutual information is symmetric  $H(X, Y) = H(Y, X)$ . Entropy is always positive, and so  $0 \leq I(X, Y) \leq \min\{H(X), H(Y)\}$ . As a result if  $X$  and  $Y$  are independent,  $0 = I(X, Y)$  [6].

Turning our attention to the case of three random variables  $X$ ,  $Y$  and  $Z$ , the Conditional Mutual Information [5, 6, 9] of  $X$  and  $Y$  given  $Z$  is then defined as,  $I(X, Y|Z) = H(X, Z) + H(Y, Z) - H(Z) - H(X, Y, Z)$  in terms of joint entropies of the random variables. It can be re-written as  $I(X, Y|Z) = H(X|Z) + H(Y|Z) - H(X, Y|Z)$ , in terms of conditional entropies or as  $I(X, Y|Z) = H(X|Z) - H(X|Y, Z)$ .

We can now define the Transfer Entropy  $T_{Y \rightarrow X}$  which is the information transfer from the distribution of random variable  $Y$  to the distribution of random variable  $X$ . This can be written as a Conditional Mutual Information

$$T_{Y \rightarrow X} = I(X_{t+1}, Y_t | X_t) = H(X_{t+1} | X_t) - H(X_{t+1} | X_t, Y_t)$$

at different time epochs  $t$  and  $t + 1$ . Here  $X_t$  and  $Y_t$  are the past states of the  $X$  and  $Y$  respectively, and  $t$  is the time index.

While the mutual information is a symmetric measurement between two variables, the transfer entropy is asymmetric measurement between two variables, as the transfer entropy represents the directional information transfer.

$$T_{Y \rightarrow X} = - \sum_{x \in X, y \in Y} p(x_{t+1}, x_t, y_t) \log_2 \frac{p(x_{t+1} | x_t, y_t)}{p(x_{t+1} | x_t)}$$

which can be reexpressed as

$$T_{Y \rightarrow X} = - \sum_{x \in X, y \in Y} p(x_{t+1}, x_t, y_t) \log_2 \frac{p(x_{t+1}, x_t, y_t)p(x_t)}{p(x_{t+1}, x_t)p(x_t, y_t)} \quad (1)$$

Transfer Entropy also can be expressed in terms of the Kullback-Leibler Divergence ( $D_{KL}$ ) as [9, 12, 15] using different time-samples. The Kullback-Leibler Divergence between two probabilistic distribution between  $p(x)$  and  $q(x)$ , as  $D_{KL}(p, q) = \sum_i p(x_i) \log_2 \frac{p(x_i)}{q(x_i)}$  [11].

Therefore, transfer entropy can be expressed as  $T_{Y \rightarrow X} = h_X - h_{XY}$ , where,

$$\begin{aligned} h_X &= - \sum_{x \in X} p(x_{t+1}, x_t) \log_2 p(x_{t+1}|x_t) = - \sum_{x \in X} p(x_{t+1}, x_t) \log_2 \frac{p(x_{t+1}, x_t)}{p(x_t)} \\ h_{XY} &= - \sum_{x \in X, y \in Y} p(x_{t+1}, x_t, y_t) \log_2 p(x_{t+1}|x_t, y_t) \\ &= - \sum_{x \in X, y \in Y} p(x_{t+1}, x_t, y_t) \log_2 \frac{p(x_{t+1}, x_t, y_t)}{p(x_t, y_t)} \end{aligned}$$

From which it is clear that,

$$\begin{aligned} h_X &= D_{KL}(p(x_{t+1}, x_t), p(x_t)) \\ h_{XY} &= D_{KL}(p(x_{t+1}, x_t, y_t), p(x_t, y_t)) \end{aligned}$$

As a result,

$$T_{Y \rightarrow X} = D_{KL}(p(x_{t+1}, x_t), p(x_t)) - D_{KL}(p(x_{t+1}, x_t, y_t), p(x_t, y_t))$$

There are a number of approaches to calculate the transfer entropy. Binning method, k-nearest neighbor method [10], or Gaussian method [13]. Each method has its own advantages or disadvantages. For instance, although the binning method is very fast, it may create a lot of empty bins or very thick bins that affects result accuracy.

## 2.2 Transfer Entropy for a Graph Edge

Suppose an edge connects node  $u$  and node  $v$ , and that associated with each node are time series  $R_u$  and  $R_v$ . For each node the time series is over a time window of duration  $\Delta t$ , and are denoted by  $R_u(t) = \{x_{t-\Delta t}^u, x_{t-\Delta t+1}^u, \dots, x_t^u\}$  and similarly  $R_v(t) = \{x_{t-\Delta t}^v, x_{t-\Delta t+1}^v, \dots, x_t^v\}$  respectively. To calculate the entropy transfer from node  $u$  to node  $v$  introduce a time delay ( $\tau$ ) for the windowed time series at node  $u$ , i.e. we consider the series  $R_u(t + \tau) = \{x_{t+\tau-\Delta t}^u, x_{t+\tau-\Delta t+1}^u, \dots, x_{t+\tau}^u\}$ . With these ingredients the entropy transfer is computable with  $R_u(t)$ ,  $R_j(t)$  and  $R_u(t + \tau)$  [4, 13].

$$\begin{aligned} T_{u \rightarrow v}(t) &= - \sum p(R_u(t + \tau), R_u(t), R_j(t)) \log_2 \frac{p(R_u(t + \tau)|R_u(t), R_v(t))}{p(R_u(t + \tau)|R_u(t))} \\ T_{u \rightarrow v}(t) &= - \sum p(R_u(t + \tau), R_u(t), R_v(t)) \log_2 \frac{p(R_u(t + \tau), R_u(t), R_v(t))p(R_u(t))}{p(R_u(t + \tau), R_u(t))p(R_u(t), R_v(t))}. \end{aligned}$$



### 3 Graphs and Transfer Entropy

Schreiber’s transfer entropy can be used to develop a new entropic characterisation of graphs derived from time series data. We use the transfer entropy to weight the edges of a graph where the nodes represent time series data and the edges represent the degree of commonality of pairs of time series. The result is a weighted graph which captures the information transfer between nodes over specific time intervals. From the weighted normalised Laplacian we characterise the network at each time interval using the von Neumann entropy computed from the normalised Laplacian spectrum, and study how this entropic characterisation evolves with time, and can be used to capture temporal changes in network structure.

To commence, we use the transfer entropy to define an edge weight  $W_{u,v}(t) = T_{u \rightarrow v}(t)$ . Suppose  $G(V, E)$  is a graph with vertex set  $V$  and edge set  $E \subseteq V \times V$  then the weighted adjacency matrix  $A$  is defined as follows

$$A(u, v) = \begin{cases} W_{u,v}, & \text{if } W_{u,v} > \textit{threshold}. \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

We have also constructed a sector graph to represent how the edge transfer entropy distributes itself across both within and between sector links. To do this suppose each node can be assigned a unique label  $\mu_u$  and that these labels can be partitioned into a set of  $m$  class-labels,  $\Omega = \{\omega_1, \dots, \omega_m\}$ . In the case of the financial data analysed later in the paper, the node labels represent individual stock, while sector labels represent different commercial or industrial sectors to which individual stock belong. With the labels to hand, we can define a weighted sector adjacency matrix, with elements

$$AT_{\omega_a, \omega_b} = \sum_{\mu_u \in \omega_a} \sum_{\mu_v \in \omega_b} W_{u,v} \tag{3}$$

The sector graph  $TG = (\Omega, AT)$  with the sector labels as nodes and weighted adjacency matrix  $AT$ . The diagonal elements are the total transfer entropy associated within individual sectors, while the off diagonal elements are the total transfer entropy between pairs of sectors.

For both graphs we need to compute the entropy. To do this we compute the normalised Laplacian matrix and from the eigenvalues of this matrix we compute the von Neumann entropy. The weighted degree matrix of graph  $G$  is a diagonal matrix  $D$  whose elements are given by  $D(u, u) = d_u = \sum_{v \in V} A(u, v)$  The normalized Laplacian matrix of the graph  $G$  is defined as  $\tilde{L} = D^{-1/2}(D - A)D^{-1/2}$  and has elements

$$\tilde{L} = \begin{cases} 1 & \text{if } u = v \text{ and } d_v \neq 0 \\ \frac{-1}{\sqrt{d_u d_v}} & \text{if } (u, v) \in E \\ 0 & \textit{otherwise} \end{cases}$$

The spectral decomposition of the normalised Laplacian matrix is  $\tilde{L} = \sum_{i=1}^{|V|} \lambda_i \phi_i \phi_i^T$  where  $\lambda_i$  are the eigenvalues and  $\phi_i$  the corresponding eigenvectors of  $\tilde{L}$ .

The von Neumann entropy was defined in quantum mechanics and can be expressed in terms of the Shannon entropy associated with the eigenvalues of the density matrix. The normalized Laplacian matrix  $\tilde{L}$  can be interpreted as the density matrix of an undirected graph [14], and the von Neumann entropy of the undirected graph can be defined as,

$$H_{VN} = - \sum_{i=1}^{|V|} \frac{\tilde{\lambda}_i}{|V|} \ln \frac{\tilde{\lambda}_i}{|V|}$$

where  $|V|$  is the number of nodes in the graph. Han et al. have shown how to approximate von Neumann entropy for undirected graph in terms of simple degree statistics using the quadratic approximation to the Shannon entropy  $x \ln x \approx x(1-x)$  [8].

$$H_{VN} \approx 1 - \frac{1}{|V|} - \frac{1}{|V|^2} \sum_{(u,v) \in E} \frac{1}{d_u d_v}$$

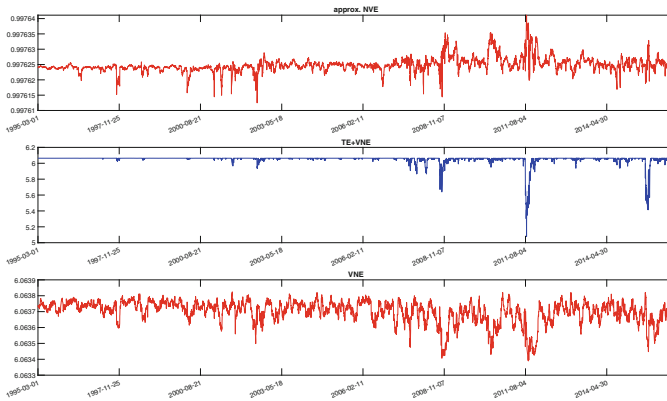
This allows the efficient calculation for the network entropy in  $O(N^2)$  rather than  $O(N^3)$  from the normalised Laplacian spectrum.

In our experiments we explore how the von Neumann entropy of the weighted graph  $G$  and the transfer entropies evolve with time for financial data covering historical stock prices. To do this we construct graphs corresponding to the trading pattern on each trading day. This yields time sequences of weighted adjacency graphs for individual stock and sector graphs for groups of stock. We represent the transfer entropy content of each graph as a long vector, and perform principal components analysis (PCA) on the time series of long-vectors. For the weighted graph  $G$  the long-vector consists of the long-vector of weighted node degree  $L = De$ , where  $e = (1, 1, 1 \dots)^T$  is the all-ones vector. For the sector graph the long-vector is a vectorisation of the upper triangle, containing both the intra-sector diagonal elements and the off-diagonal inter-sector elements. We perform PCA on these different long-vectors. We commence by computing the covariance matrix  $\Sigma$  over the complete time series, and then project the long-vectors into the space spanned by the leading eigenvectors of the covariance matrix.

## 4 Experiments

We have created a new dataset covering the closing prices of 431 companies for 5400 days on the NYSE and NASDAQ. The companies selected in this dataset come from 8 different commercial and industrial sectors, and have traded for 20 years or longer. So for example companies such as Facebook or Lehman Brothers are not listed. After we collected the data, we applied log-return ( $R_t^u = \ln(P_t^u) - \ln(P_{t-1}^u)$ , where  $P_t^u$  is the closing price of stock  $u$  on day  $t$ ) to the closing prices and use this to construct a time-series.

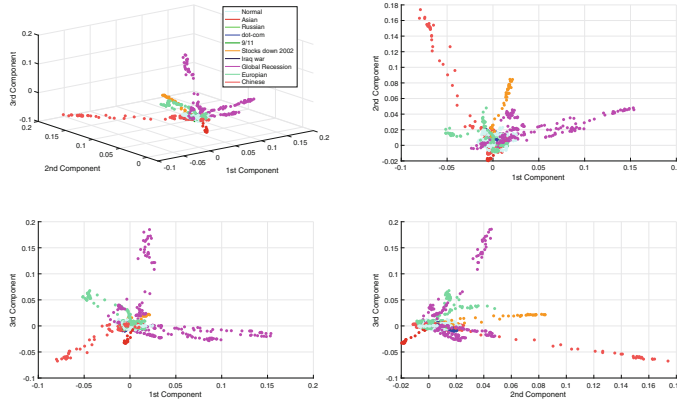
At each day of trading we construct a graph to represent the trading pattern in the markets studied. Each stock is represented by a labelled node. We compute the cross-correlation and transfer entropy between the time series for each pair of stock over a time window of 30 days. We create an edge if the cross-correlation exceeds a threshold (we choose top 5 per cent of edges according to correlation values), and attribute this edge with the transfer entropy for the time series. In addition each company traded is labelled as belonging to one of 8 different sectors. These sectors have been selected on the basis of Yahoo Finance and are as follows, Basic Material (50 stocks), Consumer Goods (62 stocks), Financial (50 stocks), Health-care (51 stocks), Industrial Goods (68 stocks), Services (49 stocks), Technology (44 stocks), Utilities (57 stocks).



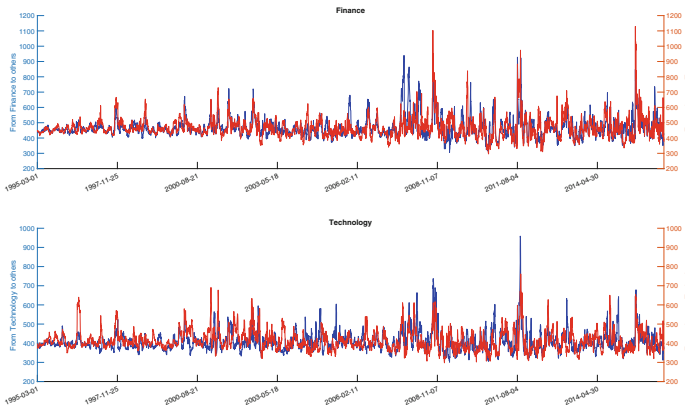
**Fig. 1.** Comparison of von Neumann entropy change with time. (Color figure online)

In Fig. 1 we show the von Neumann entropy (in blue) of the weighted transfer entropy graph as a function of time. For comparison (above in red) is the von Neumann entropy computed from the normalised Laplacian spectrum, and (below in red) is the approximate von Neumann entropy of Han et al. [8]. The main features to note are that the different financial crises emerge more clearly when we use transfer entropy to weight the edges of the graph than when the two alternatives are used. From left to right the main peaks correspond to Asian financial crisis (1997), dot-com bubble (2000), 9/11 (2001), stock market downturn (2002), global financial crisis (2007–08), European debt crisis (2009–12), Chinese stock market turbulence (2015–16).

To take this analysis of the transfer entropy one step further we perform principal components on a time series of long vectors whose components are the total transfer entropies associated with each node in the graph. In Fig. 2 we show different views of the leading three principal component projections of the long-vector time series. The different colours correspond to the financial epochs associated with different crises. It is interesting that the different crises correspond to different subspaces in the plot, following clearly clustered trajectories.



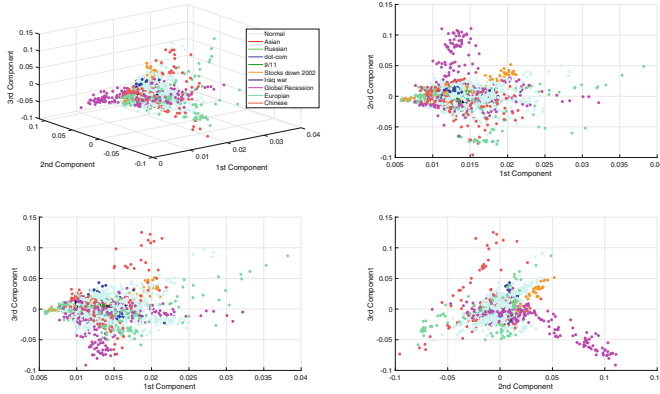
**Fig. 2.** PCA for transfer entropy stock-price graphs. (Color figure online)



**Fig. 3.** Information flow through time for the finance sector and technology sector.

In Fig. 3 we take this analysis one step further and show times series of the within and between sector transfer entropy for the finance and technology sectors. The financial sector dominates during the Global financial crisis when compared to other sectors. Moreover, it seems to be quite effective in determining the direction of the market. The technology sector, on the other hand, is generally affected by the other sectors by the middle of the 2000s. After the Dot-com bubble, it gradually moves to a position that has affected the market. In the Europe and China financial crisis, it has been observed to be passive.

Finally, in Fig. 4 we show PCA of the sector-graph. Here at each time step we construct a long-vector containing the sum of transfer entropies within and between the different sectors. We then project these long vectors onto the principal component axes for the entire time series. The plot shows different views of the three leading principal components. The different colours again represent different financial crises. The long vectors now contain just 36 upper triangular



**Fig. 4.** PCA for transfer entropy sector graphs. (Color figure online)

components rather than the 431 components for different stock, but a strong cluster structure corresponding to different crises still emerges.

## 5 Conclusion

In this paper, we have used the transfer entropy to analyse a financial market dataset covering the closing prices of stock traded over a 5400 day period. We commenced by constructing a graph in which the edges represent information flow between time series for stock, quantified using transfer entropy. The von Neumann entropy of the resulting weighted graph has been demonstrated to give a better localisation of temporal anomalies in network structure due to global financial crises. Compared to the approximate von Neumann entropy of Han et al. [8] it is less prone to noise. Moreover, PCA of the cumulative node transfer entropy with time shows that the different financial crises occupy different largely non-overlapping subspaces. Reducing the dimensionality of the problem by considering a representation based on within and between sector cumulative transfer entropy, we can still separate anomalous epochs, but less clearly.

So transfer entropy appears to capture information flow within the financial trading networks in a manner which is less prone to noise than von Neumann entropy. However, this is at the expense of computational cost.

Our future work will focus on how to use the transfer entropy representation presented in this paper to construct kernel representations of graph time series.

## References

1. Razak, F.A., Jensen, H.J.: Quantifying ‘causality’ in complex systems: understanding transfer entropy. *PLoS ONE* **9**(6), 1–14 (2014)
2. Bai, L., Hancock, E.R., Ren, P.: Jensen-Shannon graph kernel using information functionals. In: *Proceedings of the International Conference on Pattern Recognition, ICPR*, pp. 2877–2880 (2012)

3. Bai, L., Zhang, Z., Wang, C., Bai, X., Hancock, E.R.: A Graph kernel based on the Jensen-Shannon representation alignment. In: International Joint Conference on Artificial Intelligence, IJCAI, January 2015, pp. 3322–3328 (2015)
4. Barnett, L., Barrett, A.B., Seth, A.K.: Granger causality and transfer entropy are equivalent for Gaussian variables. *Phys. Rev. Lett.* **103**(23), 238701 (2009)
5. Cover, T.M., Thomas, J.A.: Entropy, relative entropy, and mutual information. In: *Elements of Information Theory*, pp. 13–55. Wiley (2005)
6. Frenzel, S., Pompe, B.: Partial mutual information for coupling analysis of multivariate time series. *Phys. Rev. Lett.* **99**(20), 1–4 (2007)
7. Granger, C.W.J.: Investigating causal relations by econometric models and cross-spectral methods. *Econometrica* **37**(3), 424 (1969)
8. Han, L., Escolano, F., Hancock, E.R., Wilson, R.C.: Graph characterizations from von Neumann entropy. *Pattern Recognit. Lett.* **33**(15), 1958–1967 (2012)
9. Hlavackovaschindler, K., Palus, M., Vejmelka, M., Bhattacharya, J.: Causality detection based on information-theoretic approaches in time series analysis. *Phys. Rep.* **441**(1), 1–46 (2007). @AssociationMeasure@
10. Kraskov, A., Stögbauer, H., Grassberger, P.: Estimating mutual information. *Phys. Rev. E - Stat. Nonlinear Soft Matter Phys.* **69**(62), 66138 (2004)
11. Kullback, S., Leibler, R.A.: On information and sufficiency. *Ann. Math. Stat.* **22**(1), 79–86 (1951)
12. Kwon, O., Yang, J.-S.: Information flow between stock indices. *EPL (Europhys. Lett.)* **82**(6), 68003 (2008)
13. Lizier, J.T.: JIDT: an information-theoretic toolkit for studying the dynamics of complex systems. *Front. Robot. AI* **1**, 11 (2014)
14. Passerini, F., Severini, S.: The von Neumann entropy of networks. In: *Developments in Intelligent Agent Technologies and Multi-Agent Systems*, pp. 66–76, December 2008
15. Schreiber, T.: Measuring information transfer. *Phys. Rev. Lett.* **85**(2), 461–464 (2000)
16. Shannon, C.E.: A mathematical theory of communication. *Bell Syst. Tech. J.* **27**(3), 379–423 (1948)
17. Smith, S.M.: Overview of fMRI analysis. In: *Functional Magnetic Resonance Imaging*, pp. 216–230. Oxford University Press, November 2001
18. Ye, C., et al.: Thermodynamic characterization of networks using graph polynomials. *Phys. Rev. E* **92**(3), 032810 (2015)
19. Ye, C., Wilson, R.C., Comin, C.H., Costa, L.D.F., Hancock, E.R.: Approximate von Neumann entropy for directed graphs. *Phys. Rev. E - Stat. Nonlinear Soft Matter Phys.* **89**(5), 52804 (2014)
20. Ye, C., Wilson, R.C., Hancock, E.R.: Graph characterization from entropy component analysis. In: *Proceedings of the International Conference on Pattern Recognition*, pp. 3845–3850. IEEE, August 2014
21. Ye, C., Wilson, R.C., Hancock, E.R.: A Jensen-Shannon divergence kernel for directed graphs. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) *S+SSPR 2016. LNCS*, vol. 10029, pp. 196–206. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_18](https://doi.org/10.1007/978-3-319-49055-7_18)



# Analyzing Time Series from Chinese Financial Market Using a Linear-Time Graph Kernel

Yuhang Jiao, Lixin Cui, Lu Bai<sup>(✉)</sup>, and Yue Wang

School of Information, Central University of Finance and Economics, Beijing, China  
bailucs@cufe.edu.cn

**Abstract.** Graph-based data has played an important role in representing complex patterns from real-world data, but there is very little work on mining time series with graphs. And those existing graph-based time series mining methods always use well-selected data. In this paper, we investigate a method for extracting graph structures, which contain the structural information that cannot be captured by vector-based data, from the whole Chinese financial time series. We call them time-varying networks, each node in these networks represents the individual time series of a stock and each undirected edge between two nodes represents the correlation between two stocks. We further review a linear-time graph kernel for labeled graphs and show whether the graph kernel, together with time-varying networks, can be used to analyze Chinese financial time series. In the experiments, we apply our method to analyze the whole Chinese Stock Market daily transaction data, i.e., the stock prices data, and use the graph kernel to measure similarities between those extracted networks. Then we compare the performances of our method and other sequence-based or vector-based methods by using kernel principle components analysis to map those results into low dimensional feature space. The experimental results demonstrate the efficiency and effectiveness of our methods together with graph kernels in analyzing Chinese financial time series.

**Keywords:** Chinese financial market · Time series · Graph kernel

## 1 Introduction

Graph-based representations are powerful tools to analyze complex real-world data. For example, Hamilton et al. [1] have used graphs to represent online social networks to predict which community the posts belong to. Li et al. [2] have adopted a graph structure to represent each video frame where the vertices denote super-pixels and the edges denote relations between these super-pixels. Wu et al. [3] have used graphs to represent the texts inside a webpage, with vertices denoting words and edges representing relations between words.

Generally speaking, there are two main advantages of using graphs. First, compared with simple structures like vectors, graphs can capture more complex features from real-world data like time series, social networks, genetic data, etc. Ignoring the structural information among those data will lead to significant information loss [11, 12], e.g., vectors can't contain the correlations between pairwise financial time series. Second, the development of kernel methods on graphs [4–6] allows us to measure the similarity between a pair of graphs efficiently [7]. Because of these benefits, a large number of works have employed graph kernels [8–10] to solve classification or clustering problems. However there is very little work on mining time series data with graph kernels, and those graph-based time series mining works always use well-selected data rather than the whole dataset to do experiments.

To overcome the aforementioned drawbacks, in this paper we propose a method for analyzing Chinese financial time series by using graph kernel. This is based on the idea that graphs can represent richer information than original data, and graph kernel can detect the significant changes of graph structure, which caused by extreme events in real-world data, effectively.

Our primary goal is to represent time series data such as financial data as graph structures, i.e., the time-varying networks, and analyze them by using a linear-time graph kernel. We commence by shifting a time window along time to construct complete weighted graphs from the original data. The nodes in the graphs are determined and labeled by the variate set of time series and connections between nodes change over time. Note that, most existing graph kernels are based on the idea of decomposing graphs into substructures and measuring pairs of isomorphic substructures [13, 14], so directly employing graph kernels to analyze such complete weighted graphs tends to be elusive. We can get the time-varying networks after reducing the number of connections between nodes. To measure the similarity of those time-varying networks, we introduce a graph kernel, i.e., Neighborhood Hash Kernel, proposed in [15], whose time complexity is related to the number of nodes times the average number of neighboring nodes in the given labeled graphs. We apply our method on the whole Chinese Stock Market data to validate the effectiveness.

The rest of the paper is organized as follows. Section 2 shows the details of how to extract the time-varying networks from multivariate time series, e.g., financial data, etc. In Sect. 3 we introduce the Neighborhood Hash kernel proposed in [15], which uses a hash function with linear time complexity. Section 4 discusses the experimental performance of our method on the whole Chinese Stock Market daily transaction data, i.e., stock closing price. Finally, in Sect. 5 we summarize our contribution present in this paper and make a suggestion for future works.

## 2 Time-Varying Network

In this section, we show the details of extracting time-varying networks from multivariate time series. Broadly speaking, the workflow of time-varying network consists of two steps, namely (a) constructing complete weighted graphs



from multivariate time series and (b) reducing the connections between nodes to extract the final form of time-varying networks. The details are as follows:

### 2.1 Complete Weighted Graph

We use a time window of size  $w$  to obtain a part of multivariate time series which contains the data over a period of  $w$ . Thus we can take each variate in this temporal window as a single vector with fixed length  $w$ . Then we create a complete weighted graph, in which each node represents a variate of the multivariate time series and the weights are determined by the Euclidean distances between those vectors, for this temporal window.

Mathematically, given a time window of size  $w$  and a set of discrete time series  $\{X_1, X_2, \dots, X_n\}$ , in which  $w$  is a positive integer and  $X_i$  represents the  $i^{th}$  variate of the multivariate time series. The distance between two variates in a temporal window at time step  $t$  can be computed as:

$$D(X_{i(t)}, X_{j(t)}) = \sqrt{\sum_{k=0}^{w-1} (x_{i(t-k)} - x_{j(t-k)})^2}, \tag{1}$$

where  $X_{i(t)} = (x_{i(t)}, x_{i(t-1)}, \dots, x_{i(t-w+1)})^T$  is the obtained vector of  $X_i$  at time step  $t$  with time window of size  $w$  and  $x_{i(t-k)}$  denotes the value of  $X_i$  at time step  $t - k$ .

By definition,  $X_{i(t)}$  and  $X_{i(t)}$  are exactly the same if and only if the distance between them is zero. On the other hand, we can tell that  $X_{i(t)}$  and  $X_{i(t)}$  are weakly related if their distance value is a large number. Also, this distance contains some time-varying information since the vector is obtained by a time window which contains the historical data. Hence, a distance matrix  $A(t)$  of those variates at time step  $t$  can be defined as:

$$A(t)_{ij} = D(X_{i(t)}, X_{j(t)}).$$

Clearly, the distance matrix  $A(t)$  is a symmetric matrix with zeros in the main diagonal. And we can take this distance matrix  $A(t)$  as the adjacency matrix of the complete weighted graph at that time step  $t$ . Then we can get a sequence of complete weighted graphs by move the time window along the whole time steps.

### 2.2 Edge Reduction

Although we have already constructed graphs containing several correlation features from multivariate time series, directly using graph kernel to measure the similarities between complete weighted graphs is still time-consuming. We have to reduce the number of connections between nodes in order to employ the kernel method more effectively. Minimum spanning tree [16] is a good choice since it selects the  $n - 1$  shortest edges from the original complete weighted graph where  $n$  is the number of nodes. Given an original weighted graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ ,

the objective function of extracting minimum spanning tree  $\mathcal{T}$  can be expressed by:

$$\min w(\mathcal{T}) = \sum_{u,v \in \mathcal{V}} w(u,v), \quad (2)$$

where  $w(u,v)$  is the weight between nodes  $u, v$ .

As we mentioned before, two nodes are considered to have strong correlation if the distance between them is short. Thus, minimum spanning trees can preserve the strongest correlation information from original graphs and reduce the edges as much as possible.

We have to do some processing on the original graphs, in order to get more potential structural information, before extracting minimum spanning trees from complete weighted graphs. Specifically, we find the shortest paths between all pairs of nodes in the graph, then we can update the adjacency matrix with the weights of all shortest paths. Fortunately, since there are many existing algorithms that can solve the all-pairs shortest path problem [17], we can simply chose one. Then, given  $SP(v_i, v_j)$  which is the weight of shortest path between nodes  $v_i$  and  $v_j$ , the updated adjacency matrix  $A'(t)$  at time step  $t$  can be:

$$A'(t)_{ij} = SP(v_i, v_j).$$

We can get a new complete weighted graph based on the updated adjacency matrix  $A'(t)$  which contains more structural information since the shortest path preserves the correlations between two nodes by considering all possible weighted path between them. Then we can extract a minimum spanning tree  $T_t$  from the new complete weighted graph at time step  $t$ , and this spanning tree is exactly the final form of the time-varying network  $G_t$ . Thus we can get a sequence of time-varying networks extracted from the multivariate time series.

### 3 Neighborhood Hash Kernel

In section, we review the Neighborhood Hash Kernel, a linear-time graph kernel, proposed by Hido et al. in [15] which maps each labeled graph into a binary array set by using a hash function. The Neighborhood Hash kernel can be simply computed by calculating the Jaccard similarity matrix, which has been proved to be a positive semi-definite matrix [18], between those binary array sets. Thus we can employ the graph kernel to measure the similarity of time-varying networks and detect the extreme events among the whole time steps efficiently.

The details of Neighborhood Hash has been introduced in [15], in order to facilitate the discussion in this paper, we make a brief review.

#### 3.1 Neighborhood Hash

Generally speaking, the Neighborhood Hash is a hash function that consists two main logical operations to map each node label into a binary array which

contains the node's neighborhood information. We commence by using a one-to-one mapping function to update the original string-like label set  $\mathcal{L}_{ori}$  into a bit-like label set  $\mathcal{L}$  which consists of binary arrays with fixed length  $D$ , the element  $l$  in set  $\mathcal{L}$  is like:

$$l = \{b_1, b_2, \dots, b_D\}, \quad (3)$$

where  $D$  satisfies  $2^D - 1 > |\mathcal{L}_{ori}|$  and  $b_i \in \{0, 1\}$ ,  $\mathcal{L}$  shares the same number of labels with  $\mathcal{L}_o$ , i.e.,  $|\mathcal{L}| = |\mathcal{L}_{ori}|$ . Now we introduce the first logical operation  $ROT$ , given a bit-like label  $l = \{b_1, b_2, \dots, b_D\}$ , the operation  $ROT$  can be:

$$ROT_o(l) = \{b_{o+1}, b_{o+2}, \dots, b_D, b_1, \dots, b_o\}, \quad (4)$$

where  $o$  is a number between 0 to  $D$ . We can tell that  $ROT$  operation changes the order of label  $l$  to get a new binary array with the same length.

Then we review the other bitwise logical operation XOR, i.e., Exclusive OR. Note that, XOR between two bits  $b_i$  and  $b_j$  gives 1 when  $b_i \neq b_j$  and 0 otherwise. Clearly, let XOR  $(l_i, l_j) = l_i \oplus l_j$ , XOR satisfies several properties:

$$\begin{aligned} l \oplus l &= l_{zero}, \\ l \oplus l_{zero} &= l, \end{aligned}$$

in which  $l_{zero}$  is a bit array full of zeros with length  $D$ , i.e.,  $l_{zero} = \{0, 0, \dots, 0\}$ .

Given a node  $v$  and its neighborhood nodes  $\{v_1^{adj}, v_2^{adj}, \dots, v_d^{adj}\}$ , we can define the Neighborhood Hash  $NH(v)$  to map  $v$ 's label  $l(v)$  into a binary array  $l'(v)$  as:

$$NH(v) = ROT_1(l(v)) \oplus l(v_1^{adj}) \oplus l(v_2^{adj}) \oplus \dots \oplus l(v_d^{adj}). \quad (5)$$

Since the hash value contains the information of neighborhood nodes, given two nodes  $v_i, v_j \in V$ , if  $NH(v_i) = NH(v_j)$ ,  $v_i$  and  $v_j$  can be considered to have the same topology except for a hash collision, whose probability of occurrence is  $2^{-D}$ .

### 3.2 Neighborhood Hash Kernel for Time-Varying Network

It is easy to compute the kernel value with the help of Neighborhood Hash. Given two labeled graphs  $\mathcal{G}_i$  and  $\mathcal{G}_j$ , we first apply the Neighborhood Hash to all of the nodes in  $\mathcal{G}_i$  and  $\mathcal{G}_j$  to obtain two new bit-like label sets  $\mathcal{L}'_i$  and  $\mathcal{L}'_j$ :

$$\begin{aligned} \mathcal{L}'_i &= \{NH(v_1), NH(v_2), \dots, NH(v_{d_i})\} \\ \mathcal{L}'_j &= \{NH(v_1), NH(v_2), \dots, NH(v_{d_j})\} \end{aligned}$$

As mentioned before, two nodes can be approximated as the same if they have the same Neighborhood Hash value, and the kernel value of  $\mathcal{G}_i$  and  $\mathcal{G}_j$  can be computed as:

$$k(\mathcal{G}_i, \mathcal{G}_j) = J(\mathcal{L}'_i, \mathcal{L}'_j), \quad (6)$$

where  $J(\mathcal{L}'_i, \mathcal{L}'_j)$  is the Jaccard similarity between  $\mathcal{L}'_i$  and  $\mathcal{L}'_j$ , then we have:

$$k(\mathcal{G}_i, \mathcal{G}_j) = \frac{|\mathcal{L}'_i \cap \mathcal{L}'_j|}{|\mathcal{L}'_i \cup \mathcal{L}'_j|} = \frac{|\mathcal{L}'_i \cap \mathcal{L}'_j|}{|\mathcal{L}'_i| + |\mathcal{L}'_j| - |\mathcal{L}'_i \cap \mathcal{L}'_j|}. \quad (7)$$

And the time complexity of this kernel is only  $O(D\bar{d}n)$  in which  $D$  is the length of bit label,  $\bar{d}$  denotes the average number of neighbors and  $n$  is the number of nodes.

In fact, there is another circumstance that two different nodes have the same Neighborhood Hash values. Considering a node  $v_i$  with three neighborhood nodes  $v_a, v_b, v_c$ , where  $l(v_a) = l(v_b)$ , the Neighborhood Hash of  $v_i$  is:

$$NH(v_i) = ROT_1(l(v_i)) \oplus l(v_a) \oplus l(v_b) \oplus l(v_c)$$

or, equivalently,

$$NH(v_i) = ROT_1(l(v_i)) \oplus l(v_c),$$

since  $l(v_a) \oplus l(v_b) = l_{zero}$ , i.e.,  $l(v_a) = l(v_b)$ , and  $l(v_c) \oplus l_{zero} = l(v_c)$ .

Now if we have another node  $v_j$  with neighborhood node  $v_d$ , and

$$l(v_i) = l(v_j),$$

$$l(v_c) = l(v_d),$$

then we can get  $NH(v_i) = NH(v_j)$ , but  $v_i$  is different from  $v_j$ . This kind of error can be avoided, and the solution has been proposed in [15].

But we don't need to take this circumstance into consideration, since our time-varying networks are extracted from multivariate time series, which nodes have unique labels. And the spanning tree algorithm ensures that each of our time-varying networks only has  $n - 1$  edges, which means the average number of neighbors  $\bar{d}$  is 1, the complexity of analyzing time-varying networks with this graph kernel is linear-time, i.e.,  $O(Dn)$ .

## 4 Experiments

In this section, we evaluate the performance of our method on a set of Chinese Stock Market data, which contains the historical transaction data of a large number of stocks. We explore whether our method can be used to analyze time series, i.e., detecting extreme financial events, effectively.

### 4.1 Dataset Preprocessing

The dataset used in this paper is extracted from Chinese Stock Market Database, which consists of the daily closing prices of 2848 stocks from December 1990 to June 2016. Due to the diversity of stock prices, we normalize the original data by calculating the closing price change ratio. Mathematically, given a stock price matrix  $S$  where  $S_{tj}$  denotes the closing price of stock  $j$  in day  $t$ , the normalized data matrix can be computed as:

$$S'_{tj} = \frac{S_{tj} - S_{t-1j}}{S_{t-1j}},$$

in particular, if the stock  $j$  has null values from day  $t_1$  to day  $t_2$  in the original data, which implies that this stock didn't open deal in those days or that stock was not existed in the market before, we set the closing price change ratio from day  $t_1$  to day  $t_2 + 1$  as 0 by default since a brand new period of trades begins on day  $t_2 + 1$ .

In this way, we can get our normalized dataset which contains the closing price change ratio of 2848 stocks from December 1990 to June 2016 (6218 days).

## 4.2 Financial Data Analysis

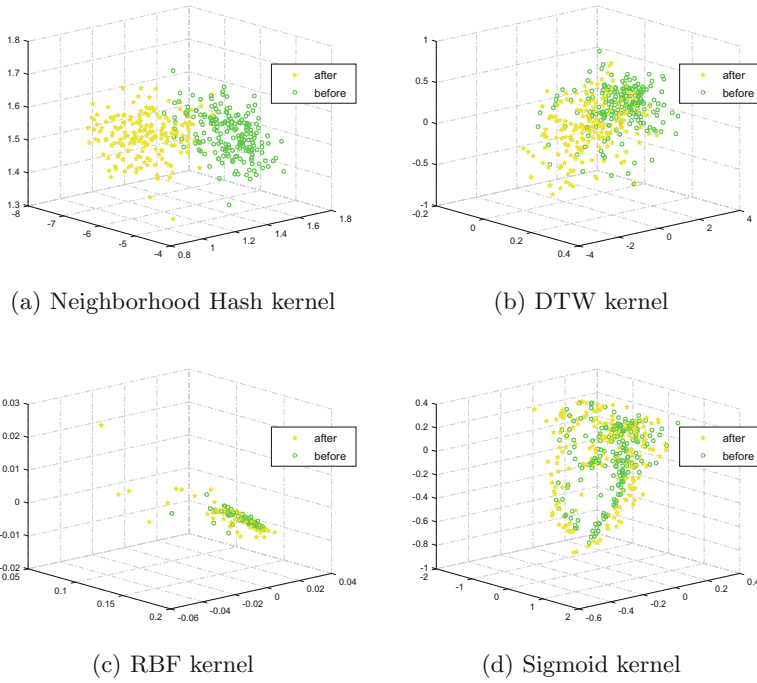
To explore the effectiveness of the proposed method for analyzing time series, i.e., detecting extreme financial events, we use a time window of 25 days and move the window along the whole time steps to extract 6194 time-varying networks and 6194 sequences from day 25 to day 6218. Each network contains the structural correlation information between 2848 stocks on one day, and each node in the network is labeled by a stock code. On the other hand, we use a 2848-dimensional vector to represent the price change ratio of 2848 stocks on one day from day 25 to day 6218. By using these methods, it is easy to obtain a network set  $G = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_{6194}\}$ , a sequence set  $S = \{\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_{6194}\}$  and a vector set  $V = \{\mathcal{V}_1, \mathcal{V}_2, \dots, \mathcal{V}_{6194}\}$  from day 25 to day 6218.

Given a kernel method with a graph set  $G$  or a sequence set  $S$  or a vector set  $V$ , we can compute a  $6194 \times 6194$  kernel matrix

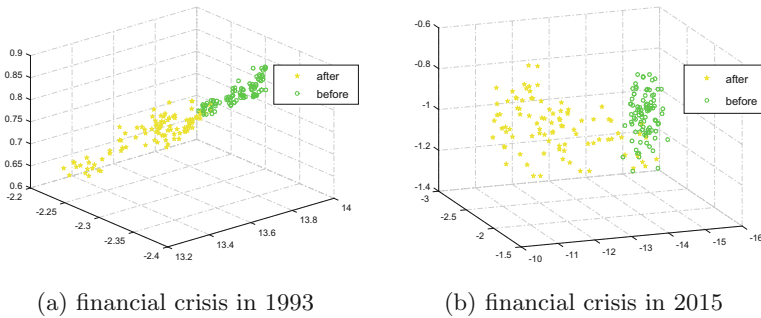
$$K = \begin{pmatrix} k_{1,1} & k_{1,2} & \cdots & k_{1,6194} \\ k_{2,1} & k_{2,2} & \cdots & k_{2,6194} \\ \vdots & \vdots & \ddots & \vdots \\ k_{6194,1} & k_{6194,2} & \cdots & k_{6194,6194} \end{pmatrix}$$

where  $k_{i,j}$  denotes the kernel value between time step  $i$  and  $j$ , e.g.,  $\mathcal{G}_i$  and  $\mathcal{G}_j$ , etc. We select a widely-used sequence kernel, i.e., Dynamic Time Warping (DTW) kernel [19], and two vector-based kernels with default parameters in open source tool scikit-learn [20], namely Radial basis function (RBF) kernel and Sigmoid kernel, to compute three different kernel matrices from sequence set  $S$  and vector set  $V$ . In order to study and visualize important features contained in the kernel matrix, we use kernel principal component analysis (Kernel PCA) [21] to embed the data to a three-dimensional principal component space.

Figure 1 shows four kernel PCA plots of kernel matrices computed from Neighborhood Hash kernel and the other three kernels during a financial crisis period in 2007. Specifically, the financial crisis started on October 16<sup>th</sup> (day 4101) and lasted for two years, so we divide 100 days before and after day 4101 into two groups. From the first plot, the embedding points separated into two distinct clusters clearly, which indicates that graph kernel has a good performance on measuring the similarity between time-varying networks. On the other hand, there are many points in different colors mixed together in those three plots, although the DTW kernel performs better than the other two kernels, which suggests that those kernels can't distinguish between these two groups well.



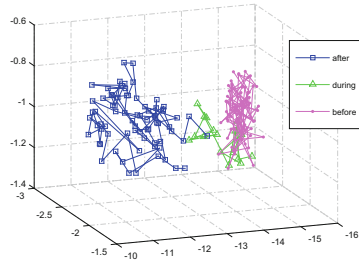
**Fig. 1.** Kernel PCA plots of four kernel methods on financial crisis data in 2007. (Color figure online)



**Fig. 2.** Kernel PCA plots of Neighborhood Hash kernel on other financial crises.

That’s because a lot of meaningful structural information has disregarded in simple structures like sequences or vectors, which, from another point of view, shows our method has great potentials in analyzing time series.

To evaluate our method better, we select the other two financial crises: (a) 100 days before and after February 16<sup>th</sup> in 1993 (day 524) and (b) 100 days before and after June 12<sup>th</sup> in 2015 (day 5964). We draw their Kernel PCA plots respectively. The result displayed in Fig. 2 also implies that our method is an



**Fig. 3.** Path of time-varying financial networks in kernel PCA space. (Color figure online)

efficient tool to analyze time series, which can simply distinguish the difference between those two groups.

What’s more, we notice that the government had promulgated a number of policies to prevent the financial crisis from getting worse in 2015, and the exact date is July 8<sup>th</sup> (day 5980) which is contained in the 100 days after day 5964. We divide the 100 days after day 5964 into two groups. The first one, noted as “during”, contains days from day 5964 to day 5980 and the other contains days after day 5980, i.e., policies promulgated date. Then, in Fig. 3, we explore the evolution of time-varying financial networks in the kernel PCA space and the experiment result is beyond our expectation. Before the financial crisis broke out, the networks represented by pink points remained stable. But the “during” group networks marked by green triangles are deviated from the pink cluster little by little. After the government promulgated policies, the networks symbolled by blue squares gradually gather into another cluster.

## 5 Conclusion

In this paper, we propose a method for extracting time-varying networks from multivariate time series automatically. In essence, the method has two steps, namely (a) generating complete weighted graphs from the time series by computing the Euclidean distance between nodes with a time window and (b) extracting minimum spanning trees from the updated complete weighted graphs whose weights are replaced by shortest paths between all pairs of nodes. Specifically, the minimum spanning trees, which contain many meaningful structural information, are the final form of time-varying networks. This extracting method, together with a linear-time graph kernel proposed in [15], allows us to analyze the time evolution of time series in a new way. In the experiments mentioned above, we have evaluated the performance of our method combined with Neighborhood Hash kernel on a set of Chinese financial data. The result clearly points the potentials of analyzing time series with graph kernels, which is more efficient than other learning techniques like sequences-based or vector-based kernel methods.

**Acknowledgments.** This work is supported by the National Natural Science Foundation of China (Grant no. 61602535, 61503422 and 61773415), the Open Projects Program of National Laboratory of Pattern Recognition, and the program for innovation research in Central University of Finance and Economics.

## References

1. Hamilton, W.L., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: *Neural Information Processing Systems*, pp. 1025–1035 (2017)
2. Li, X., et al.: Visual tracking via random walks on graph model. *IEEE Trans. Cybern.* **46**(9), 2144–2155 (2016)
3. Wu, J., et al.: Boosting for multi-graph classification. *IEEE Trans. Cybern.* **45**(3), 416–429 (2015)
4. Kashima, H.: Marginalized kernels between labeled graphs. In: *Proceedings of the Twentieth International Conference on Machine Learning*, pp. 321–328 (2003)
5. Vishwanathan, S.V.N., et al.: Graph kernels. *J. Mach. Learn. Res.* **11**(2), 1201–1242 (2008)
6. Bai, L., et al.: An aligned subtree kernel for weighted graphs. In: *International Conference on Machine Learning*, pp. 30–39 (2015)
7. Haussler, D.: Convolution kernels on discrete structures. Technical report, vol. 7, pp. 95–114 (1999)
8. Bai, L., et al.: Quantum kernels for unattributed graphs using discrete-time quantum walks. *Pattern Recognit. Lett.* **87**(C), 96–103 (2016)
9. Gärtner, T., Lloyd, J.W., Flach, P.A.: Kernels and distances for structured data. *Mach. Learn.* **57**(3), 205–232 (2004)
10. Bai, L., Hancock, E.R.: Fast depth-based subgraph kernels for unattributed graphs. *Pattern Recognit.* **50**(C), 233–245 (2016)
11. Bonanno, G., et al.: Networks of equities in financial markets. *Eur. Phys. J. B* **38**(2), 363–371 (2004)
12. Eisenberg, L., Noe, T.H.: Systemic risk in financial networks. *SSRN Electron. J.* (2007)
13. Bai, L., Escolano, F., Hancock, E.R.: Depth-based hypergraph complexity traces from directed line graphs. Elsevier Science Inc. (2016)
14. Bai, L., et al.: A quantum Jensen-Shannon graph kernel for unattributed graphs. *Pattern Recognit.* **48**(2), 344–355 (2015)
15. Hido, S., Kashima, H.: A linear-time graph kernel. In: *Ninth IEEE International Conference on Data Mining*, pp. 179–188. IEEE Computer Society (2009)
16. Prim, R.C.: Shortest connection networks and some generalizations. *Bell Labs Tech. J.* **36**(6), 1389–1401 (2013)
17. Seidel, R.: On the all-pairs-shortest-path problem. *J. Comput. Syst. Sci.* **51**(3), 400–403 (1995)
18. Gower, J.C.: A general coefficient of similarity and some of its properties. *Biometrics* **27**(4), 857–871 (1971)
19. Cuturi, M.: Fast global alignment kernels. In: *International Conference on Machine Learning*, pp. 929–936 (2011)
20. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**(10), 2825–2830 (2012)
21. Schölkopf, B., Smola, A., Mller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.* **10**(5), 1299–1319 (1998)





# A Preliminary Survey of Analyzing Dynamic Time-Varying Financial Networks Using Graph Kernels

Lixin Cui<sup>1</sup>, Lu Bai<sup>1</sup>(✉), Luca Rossi<sup>2</sup>, Zhihong Zhang<sup>3</sup>, Yuhang Jiao<sup>1</sup>,  
and Edwin R. Hancock<sup>4</sup>

<sup>1</sup> Central University of Finance and Economics, Beijing, China  
bailucs@cufe.edu.cn

<sup>2</sup> Aston University, Birmingham, UK

<sup>3</sup> Xiamen University, Fujian, China

<sup>4</sup> University of York, York, UK

**Abstract.** In this paper, we investigate whether graph kernels can be used as a means of analyzing time-varying financial market networks. Specifically, we aim to identify the significant financial incident that changes the financial network properties through graph kernels. Our financial networks are abstracted from the New York Stock Exchange (NYSE) data over 6004 trading days, where each vertex represents the individual daily return price time series of a stock and each edge represents the correlation between pairwise series. We propose to use two state-of-the-art graph kernels for the analysis, i.e., the Jensen-Shannon graph kernel and the Weisfeiler-Lehman subtree kernel. The reason of using the two kernels is that they are the representative methods of global graph kernels and local graph kernels, respectively. We perform kernel Principle Components Analysis (kPCA) associated with each kernel matrix to embed the networks into a 3-dimensional principle space, where the time-varying networks of all trading days are visualized. Experimental results on the financial time series of NYSE dataset demonstrate that graph kernels can well distinguish abrupt changes of financial networks with time, and provide a more effective alternative way of analyzing original multiple co-evolving financial time series. We theoretically indicate the perspective of developing novel graph kernels on time-varying networks for multiple co-evolving time series analysis in future work.

**Keywords:** Graph kernels · Time-varying financial networks  
NYSE dataset

## 1 Introduction

Recently, network based structure representations have been proven powerful tools to analyze multiple co-evolving time series originating from time-varying

complex systems [17,24]. This is based on the idea that the time-varying networks can well represent the interactions between the time series of system entities [7], and one can significantly analyze the system by exploring the structure variations of the networks with time. For most existing approaches, one main objective is to detect the extreme event that can significantly influence the network structures. For instance, in the financial time-varying networks abstracted from a financial market system, extreme events representing financial instability of stocks are of interest [20] and can be inferred by detecting the anomalies in the corresponding networks [23].

Generally speaking, many existing methods aim to derive network characteristics based on capturing network substructures using clusters, hubs and communities [1,2,11]. Moreover, another kind of principled approaches is to characterize the networks using ideas of statistical physics [13,14]. These methods use the partition function to describe the network, and the associated entropy, energy and temperature measures can be computed through the function [10,23]. Unfortunately, all the aforementioned methods tend to approximate network structures in a low dimensional space, and thus lead to information loss. This drawback influences the effectiveness of existing approaches for time-varying network analysis. One way to overcome this problem is to use graph kernels.

In machine learning, graph kernels are important tools for analyzing structure data represented by graphs (i.e., networks). This is because graph kernels can map graph structures in a high dimensional Hilbert space and better preserve the structure information of graphs. The most generic principle for defining a kernel between a pair of graphs is to decompose the graphs into substructures and count pairs of isomorphic substructures. Within this scenario, most graph kernels can be divided into three main categories, i.e., the graph kernels based on counting all pairs of isomorphic (a) walks [12], (b) paths [6], and (c) subgraphs or subtree structures [5,18]. Unfortunately, there are two common shortcomings arising in these substructure based graph kernels. First, these kernels cannot directly accommodate complete weighted graphs, since it is difficult to decompose a complete weighted graph into substructures. Second, these kernels tend to use substructures of limited sizes. Although this strategy curbs the notorious inefficiency of comparing large substructures, measuring kernel values with limited sized substructures only reflects local topological characteristics of a graph.

To overcome the shortcomings of the substructure based graph kernels, another family of graph kernels based on using the adjacency matrix to capture global graph characteristics have been developed by [3,15,22]. For instance, Johansson et al. [15] have developed a family of global graph kernels based on the Lovász number and its associated orthonormal representation through the adjacency matrix. Xu et al. [22] have proposed a local-global mixed reproducing kernel based on the approximate von Neumann entropy through the adjacency matrix. Bai and Hancock [3] have defined an information theoretic kernel based on the classical Jensen-Shannon divergence between the steady state random walk probability distributions obtained through the adjacency matrix. Since the adjacency matrix directly reflects the edge weighted information, these global graph kernels can naturally accommodate complete weighted graphs.

The aim of this paper is to explore whether graph kernels can be used as a means of analyzing time-varying financial market networks. Specifically, we aim to identify the significant financial incident that changes the financial network properties through graph kernels. To this end, similar to [23], we commence by establishing a family of time-varying financial networks abstracted from the New York Stock Exchange (NYSE) data over 6004 trading days, where each vertex represents the individual daily return price time series of a stock and each edge represents the correlation between pairwise series. Note that all these networks have a fixed number of vertices, i.e., these networks have the same vertex set. This is not an entirely uncommon situation, and usually arises where the time-varying networks are abstracted from complex systems having a known set of states or components. With the family of time-varying financial networks to hand, we compute the kernel matrix by measuring the graph kernel value between each pair of the networks. In this work, we propose to use two state-of-the-art graph kernels, i.e., the Jensen-Shannon graph kernel and the Weisfeiler-Lehman subtree kernel. The reason of using the two kernels is that they are the representative methods of global graph kernels and local graph kernels, respectively. We perform kernel PCA associated with each kernel matrix to embed the networks into a 3-dimensional principle space, where the time-varying networks of all trading days are visualized.

To make our investigation one step further, we compare the graph kernels with a classical dynamic time warping kernel for original time series from the NYSE dataset [8]. Moreover, we also compare the graph kernels with three classical graph characterization (embedding) methods and the visualizations are spanned by these three graph characterizations for the time-varying networks. Experimental results show that graph kernels can significantly outperform either the graph characterization method or the dynamic time warping kernel for original vectorial time series. We analyze the theoretical advantages of graph kernels on the time-varying financial network analysis, and explain the reason of the effectiveness. Our work indicates that graph kernels associated with time-varying financial networks can provide us a more effective alternative way of analyzing original multiple co-evolving financial time series.

This paper is organized as follows. Section 2 introduces the definitions of the Jensen-Shannon graph kernel and the Weisfeiler-Lehman subtree kernel. Section 3 provides the experimental results and analysis. Finally, Sect. 4 provides the conclusion.

## 2 Preliminary Concepts

In this section, we will introduce two state-of-the-art graph kernels that will be used to analyze the time-varying financial networks abstracted from NYSE dataset.

### 2.1 The Jensen-Shannon Graph Kernel

The Jensen-Shannon graph kernel [3] is based on the classical Jensen-Shannon divergence measure. In information theory, the Jensen-Shannon divergence is a

non-extensive mutual information measure defined between probability distributions [16]. Let  $\mathcal{P} = (p_1, \dots, p_m, \dots, p_M)$  and  $\mathcal{Q} = (q_1, \dots, q_m, \dots, q_M)$  be a pair of probability distributions, then the divergence measure between the distributions is

$$\begin{aligned} D_{\text{JS}}(\mathcal{P}, \mathcal{Q}) &= H_S\left(\frac{\mathcal{P} + \mathcal{Q}}{2}\right) - \frac{1}{2}H_S(\mathcal{P}) - \frac{1}{2}H_S(\mathcal{Q}) \\ &= -\sum_{m=1}^M \frac{p_m + q_m}{2} \log \frac{p_m + q_m}{2} + \sum_{m=1}^M p_m \log p_m \\ &\quad + \sum_{m=1}^M q_m \log q_m. \end{aligned} \quad (1)$$

where  $H_S(\mathcal{P}) = \sum_{m=1}^M p_m \log p_m$  are the Shannon entropies associated with  $\mathcal{P}$ . For each graph  $G(V, E)$ , we commence by computing the probability distribution of the steady state random walk visiting the vertices of  $G(V, E)$ . Specifically, the probability of the random walk on  $G(V, E)$  visiting each vertex  $v \in V$  is

$$\mathcal{P}(v) = d(v) / \sum_{u \in V} d(u), \quad (2)$$

where  $d(v)$  is the vertex degree of  $v$ . For a pair of graphs  $G_p(V_p, E_p)$  and  $G_q(V_q, E_q)$  and their associated random walk probability distributions  $\mathcal{P}$  and  $\mathcal{Q}$ , the Jensen-Shannon graph kernel  $k_{\text{JS}}(G_p, G_q)$  associated with the Jensen-Shannon divergence is

$$k_{\text{JS}}(G_p, G_q) = \exp(-D_{\text{JS}}(\mathcal{P}, \mathcal{Q})). \quad (3)$$

## 2.2 The Weisfeiler-Lehman Subtree Kernel

In this subsection, we review the concept of the Weisfeiler-Lehman subtree kernel. This kernel is based on counting the number of the isomorphic subtree pairs, as identified by the Weisfeiler-Lehman algorithm [19]. Specifically, for a sample graph  $G(V, E)$  and a vertex  $v \in V$ , we denote the neighbourhood vertices of  $v$  as  $\mathcal{N}(v) = \{u | (v, u) \in E\}$ . For each iteration  $m$  where  $m > 1$ , the Weisfeiler-Lehman algorithm strengthens the current label  $\mathcal{L}_{\text{WL}}^{m-1}(v)$  of each vertex  $v \in V$  as a new label  $\mathcal{L}_{\text{WL}}^m(v)$  by taking the union of the current labels of vertex  $v$  and its neighbourhood vertices in  $\mathcal{N}(v)$ , i.e.,

$$\mathcal{L}_{\text{WL}}^m(v) = \bigcup_{u \in \mathcal{N}(v)} \{\mathcal{L}_{\text{WL}}^{m-1}(v), \mathcal{L}_{\text{WL}}^{m-1}(u)\}, \quad (4)$$

Note that, when  $m = 1$  the current label  $\mathcal{L}_{\text{WL}}^0(v)$  of  $v$  is its initial vertex label. For each iteration  $m$  the new label  $\mathcal{L}_{\text{WL}}^m(v)$  of  $v$  corresponds to a specific subtree structure of height  $m$  rooted at  $v$ . Furthermore, for a pair of graphs  $G_p(V_p, E_p)$  and  $G_q(V_q, E_q)$ , if the new updated vertex labels of  $v_p \in V_p$  and  $v_q \in V_q$  at

the  $m$ -th iteration are identical, the subtrees corresponded by these new labels are isomorphic. Thus, the Weisfeiler-Lehman subtree kernel  $k_{\text{WL}}^{(M)}(G_p, G_q)$ , that counts the pairs of isomorphic subtrees [19], can be defined by counting the number of identical vertex labels at each iteration  $m$ , i.e.,

$$k_{\text{WL}}^{(M)}(G_p, G_q) = \sum_{m=0}^M \sum_{v_p \in V_p} \sum_{v_q \in V_q} \delta\{\mathcal{L}_{\text{WL}}^m(v_p), \mathcal{L}_{\text{WL}}^m(v_q)\}, \quad (5)$$

where

$$\delta(\mathcal{L}_{\text{WL}}^m(v_p), \mathcal{L}_{\text{WL}}^m(v_q)) = \begin{cases} 1 & \text{if } \mathcal{L}_{\text{WL}}^m(v_p) = \mathcal{L}_{\text{WL}}^m(v_q), \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

### 3 Experiments

We establish a **NYSE dataset** that consists of a series of time-varying networks abstracted from the multiple co-evolving time series of the New York Stock Exchange (NYSE) database [20, 23]. The NYSE database encapsulates daily prices of 347 stocks over 6004 trading days from January 1986 to February 2011, i.e., each of the financial network has 347 co-evolving time series of the daily return stock prices. The prices are all corrected from the Yahoo financial

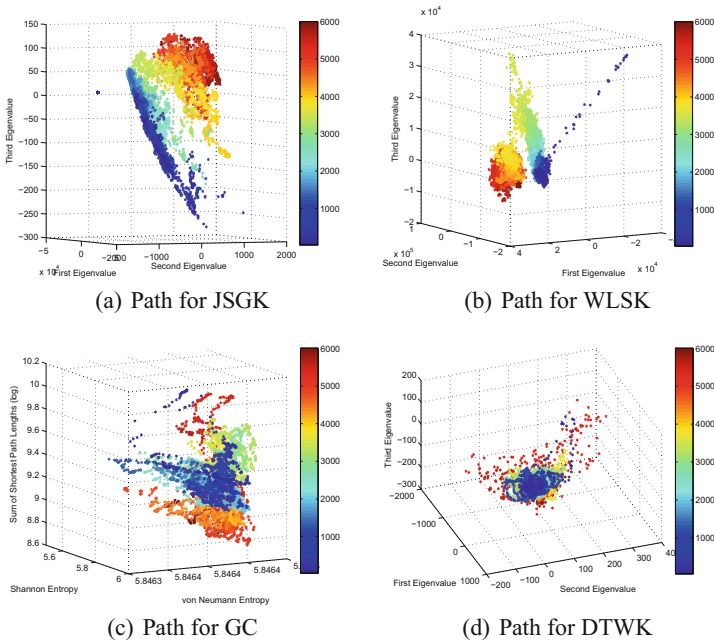
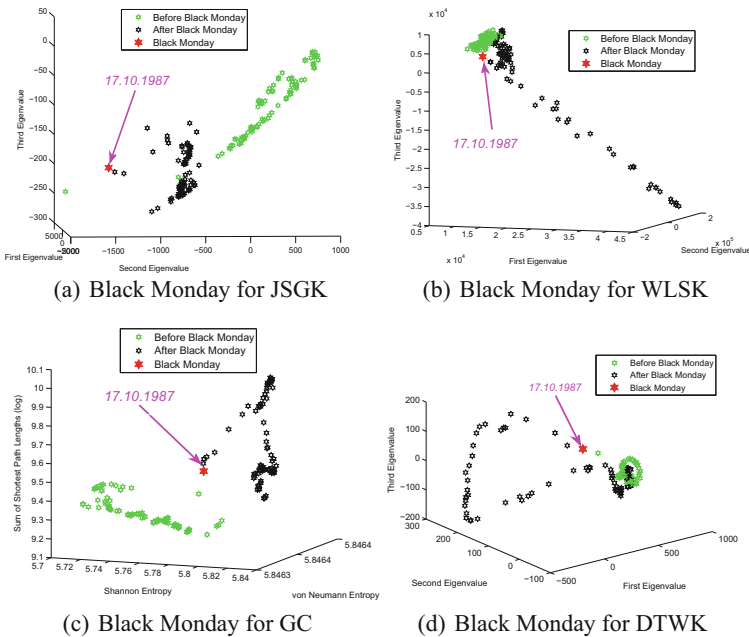


Fig. 1. Path of financial networks over all trading days. (Color figure online)

dataset (<http://finance.yahoo.com>). To extract the network representations, we use a fixed time window of 28 days and move this window along time to obtain a sequence (from day 29 to day 6004) in which each temporal window contains a time series of the daily return stock prices over a period of 28 days. We represent trades between different stocks as a network. For each time window, we compute the correlation between the time series for each pair of stocks as the weight of the connection between them. Clearly, this yields a time-varying financial market network with a fixed number of 347 vertices and varying edge weights for each of the 5976 trading days. Note that each network is a complete weighted graph. To our knowledge, the aforementioned state-of-the-art graph kernels cannot directly accommodate this kind of time-varying financial market networks, since all these kernels cannot deal with complete weighted graphs.

### 3.1 Network Visualizations from kPCA

In this subsection, we investigate whether graph kernels can be used as a means of analyzing the time-varying financial networks. Specifically, we explore whether abrupt changes in network evolution can be significantly distinguished through graph kernels. We commence by computing the kernel matrix using each of the Jensen-Shannon graph kernel (JSGK) and the Weisfeiler-Lehman subtree kernel (WLSK). Note that, the WLSK kernel cannot accommodate either complete weighted graphs or weighted graphs. Thus, we apply the WLSK kernel to the

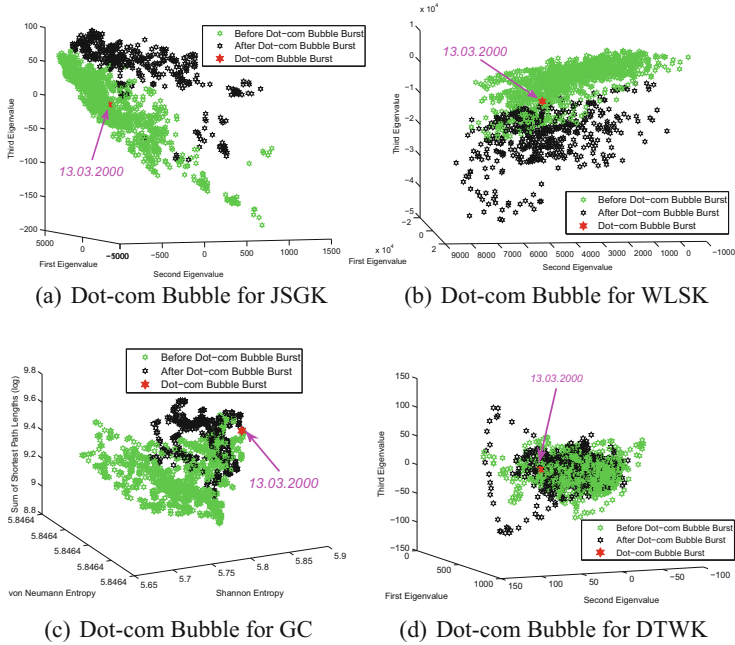


**Fig. 2.** The 3D embeddings of Black Monday. (Color figure online)

sparser un-weighted version of the financial networks, where each sparse un-weighted network is constructed by preserving only the original edges whose weights fall into the larger 10% of weights and ignoring the weights. On the other hand, the JSGK kernel can accommodate complete graphs, thus we directly perform the JSGK kernel on the original financial networks. Moreover, since each vertex label (i.e., the code of a stock represented by the vertex) appears just once for each financial network, we establish the required correspondences between a pair of networks through the vertex labels for the JSGK kernel. We perform kernel Principle Component Analysis (kPCA) [21] on the kernel matrix of the financial networks, and visualize the networks using the first three principal components in Fig. 1(a) and (b) for the JSGK and WLSK kernels respectively.

Furthermore, we compare the proposed kernels to three classical graph characterization methods (GC) that can also accommodate the original financial networks that are complete weighted graphs, i.e., the Shannon entropy associated with the steady state random walk [4], the von Neumann entropy associated with the normalized Laplacian matrix [9], and the average length of the shortest path over all pairwise vertices [20]. The visualization spanned by the three graph characterizations are shown in Fig. 1(c). Finally, we also compare the proposed kernels with the dynamic time warping kernel for original time series (DTWK) [8]. For the DTWK kernel, we also use a time window of 28 days for each trading day. We also perform kPCA on the resulting kernel matrix, and visualize the original time series using the first three principal components in Fig. 1(d). The visualization results exhibited in Fig. 1 indicate the variations of the time-varying financial networks in the different kernel or embedding spaces over 5976 trading days. The color bar beside each plot represents the date in the time series. It is clear that the results given by graph kernels form a better manifold structure.

To take our study one step further, we show in detail the visualization results during three different financial crisis periods. Specifically, Fig. 2 corresponds to the Black Monday period (*from 15th Jun 1987 to 17th Feb 1988*), Fig. 3 to the Dot-com Bubble period (*from 3rd Jan 1995 to 31st Dec 2001*), and Fig. 4 to the Enron Incident period (*the red points, from 16th Oct 2001 to 11th Mar 2002*). Figures 2, 3 and 4 indicate that Black Monday (*17th Oct, 1987*), the Dot-com Bubble Burst (*13rd Mar, 2000*), and the Enron Incident period (*from 2nd Dec 2001 to 11th Mar 2002*) are all crucial financial events, since the network embedding points through the kPCA of the JSGK and WLSK kernels form two obvious clusters before and after the event. In other words, the JSGK and WLSK graph kernels can well distinguish abrupt changes in network evolutions with time. Another interesting feature in Fig. 4 is that the networks between 1986 and 2011 are separated by the Prosecution against Arthur Andersen (*3rd Nov, 2002*). The prosecution is closely related to the Enron Incident. As a result, the Enron Incident can be seen as a watershed at the beginning of 21st century, that significantly distinguishes the financial networks of the 21st and 20th centuries. On the other hand, the GC method and the DTWK kernel on original time series can only distinguish the financial event of Black Monday, and fail to distinguish other events.



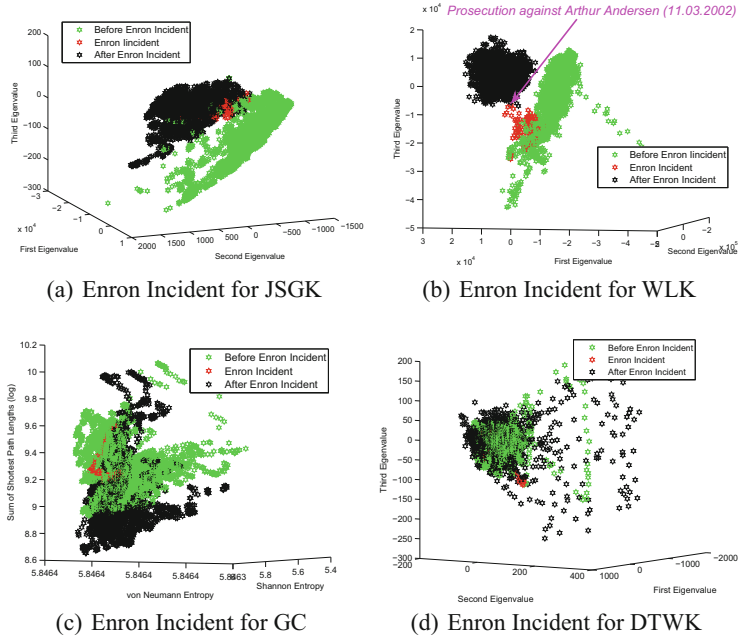
**Fig. 3.** The 3D embedding of Dot-com Bubble Burst. (Color figure online)

### 3.2 Experimental Analysis

The above experimental results demonstrate that graph kernels can be powerful tools for analyzing time-varying financial networks. The reasons of the effectiveness are twofold. First, unlike the original multiple co-evolving time series from the NYSE dataset, the abstracted time-varying financial networks can reflect rich co-related interactions between the original time series. Second, the graph kernels can map network structures in a high dimensional Hilbert space, and thus better preserve the structure information of original time series encapsulated in the networks. By contrast, the GC method can also directly capture network characteristics. However, as one kind of graph embedding methods, the GC method tends to approximate the network structures in low dimensional space and leads to information loss. On the other hand, although the DTWK kernel can map the original time series in a high dimensional Hilbert space, the DTWK kernel on original time series cannot directly capture the co-related interactions between the time series. These observations demonstrate that graph kernels associated with time-varying financial networks can provide us a more effective alternative way of analyzing original multiple co-evolving financial time series.

Although both the JSGK and WLSK graph kernels can well distinguish the abrupt changes of financial networks with time. We can also observe some different phenomenons between the kPCA embeddings through the two graph kernels.





**Fig. 4.** The 3D embedding of Enron Incident. (Color figure online)

For instance, Fig. 1 indicates that the embedding points through the WLSK kernel can form a better transiting with time than the JSGK kernel, when we visualize all the financial networks over the 6004 trading days. Moreover, Fig. 4 also visualizes all the financial networks and the kPCA embeddings through the WLSK kernel form better clusters before and after the Enron incident than the JSGK kernel. This may be caused by the fact that the WLSK kernel is performed on the sparser version of the original time-varying financial networks, i.e., the edges corresponding to lower co-relations between pairwise time-series represented by vertices are deleted. As a result, the WLSK kernel can capture the dominant co-related information between pairwise time series, and ignore the noises accumulated from the lower co-relations over all the 6004 trading days. By contrast, although the JSGK kernel can completely capture all the information through the original financial networks that are complete graphs, its effectiveness may be also influenced by the lower co-relations with noises. On the other hand, Figs. 3 and 2 indicate that sometimes the JSGK kernel can form more separated clusters than the WLSK kernel, when we only visualize the financial networks over a small number of trading days around the financial event. This may be caused by the fact that only the JSGK kernel can accommodate the complete network structures and reflect global network characteristics. Moreover, the effect of the lower co-related information between time series over a small number of trading days may be minor and will not seriously influence the effectiveness.

The above observations indicate that how to balance the trade off between capturing global complete network structures and eliminating noises through sparser network structures is important for developing new graph kernels in future works. Finally, note that, although the time-varying financial networks can reflect richer co-relations between pairwise time series, these networks inevitably lost the original time series information. One way to overcome this problem is to associate the original vectorial time series to each corresponding vertex as the vectorial continuous vertex label. Unfortunately, neither of the JSGK and the WLSK graph kernels can accommodate such kind of vertex labels. Developing approaches of accommodating vectorial continuous vertex labels may be an inspired way of developing novel graph kernels on time-varying networks for multiple co-evolving time series analysis in future work.

## 4 Conclusion

In this paper, we have investigated that graph kernels are powerful tools of analyzing time-varying financial market networks. Specifically, we have established a family of time-varying financial networks abstracted from the New York Stock Exchange data over 6004 trading days. Experimental results have demonstrated that graph kernels can not only well distinguish abrupt changes of financial networks with time, but also provide a more effective alternative way of analyzing original multiple co-evolving financial time series. Finally, we theoretically indicate the perspective of developing novel graph kernels on time-varying network analysis for future work.

**Acknowledgments.** This work is supported by the National Natural Science Foundation of China (Grant no. 61602535, 61503422 and 61773415), the Open Projects Program of National Laboratory of Pattern Recognition, and the program for innovation research in Central University of Finance and Economics.

## References

1. Anand, K., Bianconi, G., Severini, S.: Shannon and von neumann entropy of random networks with heterogeneous expected degree. *Phys. Rev. E* **83**(3), 036109 (2011)
2. Anand, K., Krioukov, D., Bianconi, G.: Entropy distribution and condensation in random networks with a given degree distribution. *Phys. Rev. E* **89**(6), 062807 (2014)
3. Bai, L., Hancock, E.R.: Graph kernels from the Jensen-Shannon divergence. *J. Math. Imaging Vis.* **47**(1–2), 60–69 (2013)
4. Bai, L., Rossi, L., Torsello, A., Hancock, E.R.: A quantum Jensen-Shannon graph kernel for unattributed graphs. *Pattern Recogn.* **48**(2), 344–355 (2015)
5. Bai, L., Rossi, L., Zhang, Z., Hancock, E.R.: An aligned subtree kernel for weighted graphs. In: *Proceedings of ICML*, pp. 30–39 (2015)
6. Borgwardt, K.M., Kriegel, H.-P.: Shortest-path kernels on graphs. In: *Proceedings of the IEEE International Conference on Data Mining*, pp. 74–81 (2005)

7. Bullmore, E., Sporns, O.: Complex brain networks: graph theoretical analysis of structural and functional systems. *Nat. Rev. Neurosci.* **10**(3), 186–198 (2009)
8. Cuturi, M.: Fast global alignment kernels. In: *Proceedings of ICML*, pp. 929–936 (2011)
9. Dehmer, M., Mowshowitz, A.: A history of graph entropy measures. *Inf. Sci.* **181**(1), 57–78 (2011)
10. Delvenne, J.-C., Libert, A.-S.: Centrality measures and thermodynamic formalism for complex networks. *Phys. Rev. E* **83**(4), 046117 (2011)
11. Feldman, D.P., Crutchfield, J.P.: Measures of statistical complexity: why? *Phys. Lett. A* **238**(4), 244–252 (1998)
12. Gärtner, T., Flach, P., Wrobel, S.: On graph kernels: hardness results and efficient alternatives. In: Schölkopf, B., Warmuth, M.K. (eds.) *COLT-Kernel 2003*. LNCS (LNAI), vol. 2777, pp. 129–143. Springer, Heidelberg (2003). [https://doi.org/10.1007/978-3-540-45167-9\\_11](https://doi.org/10.1007/978-3-540-45167-9_11)
13. Huang, K.: *Statistical Mechanics*. Wiley, New York (1987)
14. Javarone, M.A., Armano, G.: Quantum-classical transitions in complex networks. *J. Stat. Mech: Theory Exp.* **2013**(04), 04019 (2013)
15. Johansson, F.D., Jethava, V., Dubhashi, D.P., Bhattacharyya, C.: Global graph kernels using geometric embeddings. In: *Proceedings of ICML*, pp. 694–702 (2014)
16. Martins, A.F.T., Smith, N.A., Xing, E.P., Aguiar, P.M.Q., Figueiredo, M.A.T.: Nonextensive information theoretic kernels on measures. *J. Mach. Learn. Res.* **10**, 935–975 (2009)
17. Nicolis, G., Cantu, A.G., Nicolis, C.: Dynamical aspects of interaction networks. *Int. J. Bifurcat. Chaos* **15**, 3467 (2005)
18. Shervashidze, N., Vishwanathan, S.V.N., Mehlhorn, K., Petri, T., Borgwardt, K.M.: Efficient graphlet kernels for large graph comparison. *J. Mach. Learn. Res.* **5**, 488–495 (2009)
19. Shervashidze, N., Schweitzer, P., van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-Lehman graph kernels. *J. Mach. Learn. Res.* **12**, 2539–2561 (2011)
20. Silva, F.N., Comin, C.H., Peron, T.K., Rodrigues, F.A., Ye, C., Wilson, R.C., Hancock, E.R., Costa, L.D.F.: Modular dynamics of financial market networks. *arXiv preprint arXiv:1501.05040* (2015)
21. Witten, I.H., Frank, E., Hall, M.A.: *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, Los Altos (2011)
22. Xu, L., Niu, X., Xie, J., Abel, A., Luo, B.: A local-global mixed kernel with reproducing property. *Neurocomputing* **168**, 190–199 (2015)
23. Ye, C., Comin, C.H., Peron, T.K., Silva, F.N., Rodrigues, F.A., Costa, L.F., Torsello, A., Hancock, E.R.: Thermodynamic characterization of networks using graph polynomials. *Phys. Rev. E* **92**(3), 032810 (2015)
24. Zhang, J., Small, M.: Complex network from pseudoperiodic time series: topology versus dynamics. *Phys. Rev. Lett.* **96**, 238701 (2006)



# Few-Example Affine Invariant Ear Detection in the Wild

Jianming Liu<sup>1</sup>(✉), Yongsheng Gao<sup>2</sup>, and Yue Li<sup>2</sup>

<sup>1</sup> School of Computer Science and Engineering, Jiangxi Normal University, Nanchang, China

liujianming@zju.edu.cn

<sup>2</sup> School of Engineering, Griffith University, Nathan Campus, Brisbane, Australia

yongsheng.gao@griffith.edu.au

**Abstract.** Ear detection in the wild with the varying pose, lighting, and complex background is a challenging unsolved problem. In this paper, we study affine invariant ear detection in the wild using only a small number of ear example images and formulate the problem of affine invariant ear detection as a task of locating an affine transformation of an ear model in an image. Ear shapes are represented by line segments, which incorporate structural information of line orientation and line-point association. Then a novel fast line based Hausdorff distance (FLHD) is developed to match two sets of line segments. Compared to existing line segment Hausdorff distance, FLHD is one order of magnitude faster with similar discriminative power. As there are a large number of transformations to consider, an efficient global search using branch-and-bound scheme is presented to locate the ear. This makes our algorithm be able to handle arbitrary 2D affine transformations. Experimental results on real-world images that were acquired in the wild and Point Head Pose database show the effectiveness and robustness of the proposed method.

**Keywords:** Ear location · Affine invariant · Branch-and-bound

## 1 Introduction

Ear biometric has gained much attention in the recent years. Most of the ear biometric techniques have focused on recognizing manually cropped ears. However, effective and robust ear detection techniques are the key component of automatic ear recognition systems. There have been some research works on the ear detection [2, 4–10]. Most of the existing works are limited to laboratory-like setting that the images are acquired under controlled condition. The problem of ear detection in uncontrolled environments is still challenging, especially using a small number of samples, as ear image may vary in shapes, sizes and colors under various viewing conditions.

---

This work was financially supported by the Natural Science Foundation of China (No. 61662034), the Youth Science Foundation of Education Department of Jiangxi Province (No. 150353) and China Scholarship Council (CSC) Scholarship (No. 201609470005).

In this work, we try to address the gap. Our work is based on the following fact: when the scale of the object is relatively small in comparison to its distance to the camera, the group of affine transformation is a good approximation of the perspective projection [1]. We formulate the ear detection in the wild as a task of locating an affine transformation of an ear model in an image. Different from traditional methods that use points to represent ear shapes [2], we represent the ear shapes using a set of line segments, which not only have efficient storage capability, but also incorporate structural information of line orientation and line-point association. Moreover, we offer a fast line segment Hausdorff distance (FLHD) to compute the similarity of two sets of line segments. Compared to existing line segment Hausdorff distance [3, 17], FLHD is one order of magnitude faster with similar discriminative power. As there are a huge number of transformations to consider, an efficient global search in affine transformation space using branch-and-bound scheme is presented to locate the ear. This makes our method be able to handle arbitrary 2D affine transformations. Our approach not only gives the location information of ear, but also can estimate the poses of ears.

### 1.1 Related Works

In this section, we review the most important techniques for ear detection. The first well-known technique for ear detection is introduced by Berge et al. [4], which depends on building neighborhood graph from the deformable contours of ears. However, it needs user interaction and is not fully automatic. In [5], the authors propose a force field technique to locate the ear. However, it only works in simple background. Prakash and Gupta [6] make use of the connected components in a graph obtained from the edge map of the side face image to locate ear's area. Experimental results depend on quality of the input image and proper illumination conditions. The ear detection method in [7] uses features from texture and depth images, as well as context information for detecting ears. The authors of [8] present an entropy-cum-Hough-transform based approach for enhancing the performance of an ear detection system. A combination of a hybrid ear localizer and an ellipsoid ear classifier is used to predict locations. In [2], an automated ear location technique based on the template matching with modified Hausdorff distance is proposed. It is invariant to illumination and occlusion in profile face image. However, it is not invariant to the rotation. All of above methods are limited to controlled image acquisition conditions and are not invariant to affine transformation. Recently, some deep learning-based ear detections are proposed [9, 10]. In [9], the problem of ear detection was formulated as a two-class segmentation problem and a convolutional encoder-decoder network based on the SegNet architecture was trained to distinguish between image-pixels belonging to either the ear or the non-ear class. However, deep learning based methods need a huge number of training samples containing all the possible situations.

## 2 Line Based Ear Model and Matching

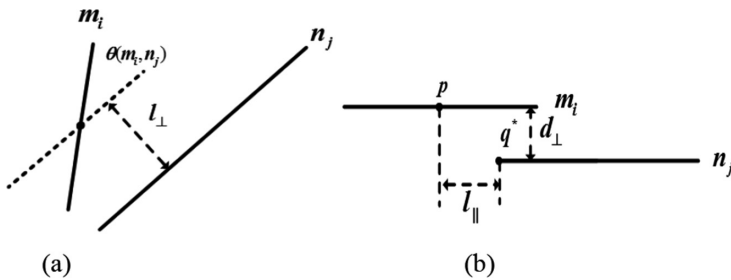
In this section, we first introduce the creation of a common ear template, and then define the distance between two line-segments. Finally, a fast line segment Hausdorff distance (FLHD) is proposed to match ear model and target image.

### 2.1 Ear Template Generation

A good ear template should incorporate various ear shapes. Human ear can broadly be grouped into four kinds: triangular, round, oval, and rectangular [2]. In this paper, we select a few ear images manually by taking above mentioned types of ear shapes into consideration. Edge detection and line segment fitting are carried out on each kind of ear images [14]. The ear edge template is generated by averaging shapes of four kinds of ears.

### 2.2 Distance Between Two Line Segments

After edge detection and line segment fitting, ear template and input target image can be represented by two sets of line segments  $M = \{m_1, m_2, \dots, m_l\}$  and  $I = \{n_1, n_2, \dots, n_k\}$ . Then ear detection problem is converted to the matching of two sets of line segments. To compare two line segments, three aspects of difference should be considered [3]: perpendicular distance ( $d_{\perp}$ ), parallel distance ( $d_{\parallel}$ ) and orientation distance ( $d_{\theta}$ ), as shown in Fig. 1.



**Fig. 1.** The distance between two line-segments. (a) The perpendicular distance  $d_{\perp}$  and orientation distance  $d_{\theta}$ . (b) The parallel distance ( $d_{\parallel}$ ).

- **perpendicular distance:**  $d_{\perp}$  is simply the vertical distance  $l_{\perp}$  between two line-segments.
- **parallel distance:** ( $d_{\parallel}$ ) is the displacement to align two parallel line-segments. As a line-segment in the target image may correspond to multiple line segments in the template (the resolution of target image is usually lower than the template, more line segments will be fitted out on the high-resolution image with same threshold), or some target lines may be partial occluded. In order to alleviate the effects of fragmentation and partial occlusion, we define it as the minimum displacement to align any points on a target line-segment  $n_j$  to the middle point of a model line-segment  $m_i$ .

$$(d_{\parallel})(m_i, n_j) = \min_{q \in n_j} (l_{\parallel})(q, m_i) \tag{1}$$

- **orientation distance:**  $d_\theta$  computes the smallest intersecting angle between  $m_i$  and  $n_j$ , which is defined as:

$$d_\theta = \min(|\theta_{m_i} - \theta_{n_j}|, ||\theta_{m_i} - \theta_{n_j}| - \pi|) \quad (2)$$

where  $\theta \in [0, \pi)$  is line segment direction angle and computed at modulo  $\pi = 180^\circ$ .

In general,  $m_i$  and  $n_j$  would not be in parallel. We can rotate the model line-segment with its mid-point as rotation center before the computation of  $d_\perp$  and  $(d_\parallel)$ . Then, the distance between two line-segments is defined as

$$d(m_i, n_j) = \sqrt{d_\parallel^2(m_i, n_j) + d_\perp^2(m_i, n_j)} + w_o \cdot d_\theta \quad (3)$$

where  $w_o$  is the weight for orientation distance and would be determined by a training process. Suppose  $p_i$  is the middle point of  $m_i$ , then we have

$$d(m_i, n_j) = \sqrt{\min_{q \in n_j} l_\parallel^2(q, m_i) + d_\perp^2} + w_o \cdot d_\theta = \min_{q \in n_j} d(p_i, q) + w_o \cdot d_\theta \quad (4)$$

where  $d(p, q)$  is the Euclidean distance between two points. Based on above definition, the computation of FLHD built on it can be speed up with 3-dimension distance transform.

### 2.3 Fast Line Segment Hausdorff Distance

The Hausdorff distance is a typical measure for shape comparison and widely used in the field of 2D and 3D point set matching [11]. Dubuisson and Jain [12] investigated 24 forms of different Hausdorff distance and indicated that a modified Hausdorff distance (MHD) gave the best performance.

Based on MHD, a directed line segment Hausdorff distance (LHD) is introduced to eliminate the outlier of line segments. It is defined as

$$h(M, I) = \frac{1}{\sum_{m_i \in M} l_i} \sum_{m_i \in M} l_i \cdot \min_{n_j \in I} d(m_i, n_j) \quad (5)$$

where  $l_i$  is the length of the model line segment  $m_i$ . The complexity of LHD is  $O(k_l N_m N_I)$ , where  $N_m$  is the number of line segments in  $M$ ,  $N_I$  is the number of line segments in the target image  $I$ , and  $k_l$  is the time to compute  $d(m_i, n_j)$ . To accelerate the computation of the LHD, a 3-dimension weighted Euclidean distance transform of a line edge image is used, which defined as

$$\Delta(x, y, \theta) = \min_{n_i \in I} (\min_{q \in n_i} d((x, y), q) + w_o \cdot d_\theta(\theta, \theta_{n_i})) \quad (6)$$

where  $x$  and  $y$  are bounded by the image dimension and  $\theta \in [0, \pi]$ .  $d((x, y), q)$  is the Euclidean distance between point  $(x, y)$  and  $q$ .  $\Delta$  can be computed in linear time [13].

Suppose a model line segment  $m_i$  are represented by 4-dimension vector  $(x_i, y_i, \theta_i, l_i)$ , where  $(x_i, y_i)$  is the mid-point coordinates of  $m_i$ ,  $\theta_i$  is the direction angle and  $l_i$  is the length of  $m_i$ . Then, we can get the FLHD as

$$\begin{aligned}
 h_f(M, I) &= \frac{1}{\sum_{m_i \in M} l_i} \sum_{m_i \in M} l_i \cdot \min_{n_j \in I} d(m_i, n_j) = \\
 &\frac{1}{\sum_{m_i \in M} l_i} \sum_{m_i \in M} l_i \cdot \min_{n_j \in I} \min_{q \in n_j} (d(p, q) + w_o \cdot d_\theta) = \\
 &\frac{1}{\sum_{m_i \in M} l_i} \sum_{m_i \in M} l_i \cdot \Delta(x_i, y_i, \theta_i)
 \end{aligned} \tag{7}$$

Given the array  $\Delta$ ,  $h_f(M, I)$  can be computed in  $O(N_m)$  pass through  $\Delta$ .

### 3 Efficient Transform Space Search for Ear Detection

Given ear model and target image encoded into line segment sets, affine invariant ear detection can be formulated as locating an affine transformation  $t$  that comes to minimize the  $h_f(M, I)$ . For any transformation  $t \in T$ , we assume a quality function as

$$f : T \rightarrow \mathbb{R} \tag{8}$$

where  $T$  is the set of 2D affine transformations of the plane.  $f(t) = -h_f(t(M), I)$  is the quality of the prediction that an ear is located at the transformation  $t$ . To predict the best location of the ear, we have to solve

$$t_{opt} = \operatorname{argmax}_{t \in T} f(t) \tag{9}$$

Exhaustively examining all affine transformations is prohibitively expensive to perform. In the following, we propose an efficient affine transform space search (ETSS) algorithm, which relies on a branch-and-bound scheme.

#### 3.1 Branch-and-Bound Scheme

To increasing the efficiency of the transform space search, we discretize the space  $T$  of affine transform by dividing each of the dimensions into  $\Theta(\delta)$  equal segments and split the transformation space into a list of non-overlap cells. A cell  $T_i$  is a rectilinear axis-aligned region of six-dimension transformation space. We parameterize  $T_i$  by its center point and the radius from the center point in each dimension. This allows the efficient representation of affine cells as  $T_i = \{t_i, r_i\}$ . The optimization works by hierarchically splitting the cells into disjoint sub-cells. For each cell, the upper and lower bounds are determined. Promising parts of cells with high upper bound are explored first, and large parts of cells do not have to be examined further if their upper bound indicates that they cannot contain the maximum.

The lower bound  $f_{lo}(T_i)$  is defined as the  $f(t_i)$  provided by the center transformation  $t_i$  of a cell. It is an estimation of the similarity provided by the current cell. We also store the largest value of  $f_{lo}(T_i)$  as the best similarity  $f_{best}$  and its associated transformation  $t_{best}$  as



best transform estimation.  $f_{up}(T_i)$  is the maximum similarity that can probably be obtained for any transformation sampled from a cell. Algorithm 1 gives the pseudo-code.

---

### Algorithm 1. Efficient Transform Space Search

---

Input:  $M, I$ ; an initial cell  $T$ , which is assumed to contain the optimum transformation;  $f_{best} = -\infty, \epsilon_a$ ;

Output: A transformation  $t^*$ .

1. Create a list of cells that cover  $T$  and do not overlap each other.
  2. Compute each cell's lower and upper bounds. Set  $f_{best} = \max_{T_k \in T}(f_{lo}(T_k))$ ,  $t_{best}$  be the associated transformation. Initializing  $P$  as empty priority queue.
  3. **For** each cell  $T_k$  in the list, **if**  $f_{up}(T_k) > f_{best}$ , push  $T_k$  into  $P$ , else kill it.
  4. **If**  $f_{best} > \epsilon_a$  or  $P$  is empty, terminate the search, and return  $f_{best}$  and  $t_{best}$ ; Else pop the top cell  $T^*$  from  $P$ .
  5. Decomposing the cell  $T^*$  into  $2^6$  disjoint cells via dividing each dimension by 2. Such that  $T^* = T_1 \cup T_2 \cup \dots \cup T_{64}$ .
  6. **For** each  $T_i$  ( $i = \{1, 2, \dots, 64\}$ ):
    - a. Compute  $T_i$ 's lower and upper bounds
    - b. **If**  $f_{lo}(T_i) > f_{best}$ , update  $f_{best}$  and  $t_{best}$ .
    - c. **If**  $f_{up}(T_i) > f_{best}$ , push  $T_i$  to the priority queue  $P$ , else kill the cell
  7. **Return** to step 4.
- 

### 3.2 Fast Estimation of Similarity Bounds

The upper similar bound is the key to the branch-and-bound search. The tighter upper bound we get, the more efficient branch-and-bound search will be. Suppose a model line segment  $m_i$  is represented by its end-points  $(p_{i,1}, p_{i,2})$ , and  $T_k(m_i) = (T_k(p_{i,1}), T_k(p_{i,2}))$  be the transformed line segments of  $m_i$  under any transform in cell  $T_k$ , as shown in Fig. 2.  $T_k(p_{i,1})$  and  $T_k(p_{i,2})$  associate with two uncertain regions,  $Br(p_{i,1}, T_k)$  and  $Br(p_{i,2}, T_k)$ . Each uncertain region corresponds to a bounding rectangle which contains all possible positions of the line segment's end-points under transformations in cell  $T_k$ . The mid-points  $p_{ij}^* = (x_{ij}^*, y_{ij}^*)$ ,  $j = 1, 2$  of  $Br(p_{ij}, T_k)$  are the transformed end-points of model line segment under the mid-transform  $t^*$ . Using the transform parameters defined in the cell  $T_k$ , the width  $w_{ij}$  and height  $h_{ij}$  of  $Br(p_{ij}, T_k)$ ,  $j = 1, 2$  can be calculated as

$$w_{ij} = 2 \cdot (r_{11}^k \cdot |x_{ij}^*| + r_{12}^k \cdot |y_{ij}^*| + r_{13}^k) \quad (10)$$

$$h_{ij} = 2 \cdot (r_{21}^k \cdot |x_{ij}^*| + r_{22}^k \cdot |y_{ij}^*| + r_{23}^k) \quad (11)$$

As the end-points' positions of transformed line segment just can change in the  $Br(p_{i,j}, T_k)$ , the maximum angle  $\theta_{max}$  and minimum angle  $\theta_{min}$  of the transformed line segment can be easily computed using the end-points of  $Br(p_{i,j}, T_k)$ , as illustrated in Fig. 2. Before computing the upper similar bound, we define a three-dimension box distance transform as

$$\Delta_{wh\theta}[x, y, \theta] = \min_{\substack{-w/2 \leq \Delta x \leq w/2 \\ -h/2 \leq \Delta y \leq h/2 \\ \theta_{min} \leq \theta \leq \theta_{max}}} \Delta(x + \Delta x, y + \Delta y, \theta) \quad (12)$$

Given the 3D distance transform array  $\Delta$ ,  $\Delta_{wh\theta}[x, y, \theta]$  can be computed in constant time by using some prefix techniques [15]. As the mid-point of the transformed line segment  $T_k(m_i)$  can only change in the related uncertain region  $Br(p_i, T_k)$ , we can get the upper bound by searching the minimum in  $Br(p_i, T_k)$ . Suppose  $t \in T_k$ ,  $p_i^t = (x_i^t, y_i^t, \theta_i^t)$  is the mid-point of the transformed line segment  $t(m_i)$ , we have

$$f(t) = -\frac{1}{\sum_{m_i \in M} l_i} \sum_{m_i \in M} l_i \cdot \Delta(x_i^t, y_i^t, \theta_i^t) \leq -\frac{1}{\sum_{m_i \in M} l_i} \sum_{m_i \in M} l_i \cdot \Delta_{w_i h_i \theta_i^t}[x_i^t, y_i^t, \theta_i^t] \quad (13)$$

where  $w_i$  and  $h_i$  are the width and height of  $Br(p_i, T_k)$ , which can be computed using Eqs. (10) and (11).  $\theta_{min} \leq \theta_i^t \leq \theta_{max}$ .

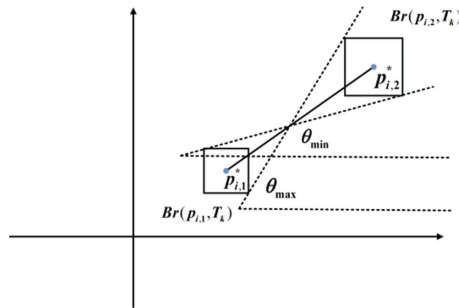


Fig. 2. Fast estimation of similar bounds.

### 4 Experimental Results

In our experiments, we evaluated our method on two datasets: Head pose database [16], and our own dataset (WildEar). The hardware used for experiment is a desktop PC with Intel® Core™ I7-3770K CPU with 16 GB system memory. The orientation angle of a line segment is quantified into 180 bins. To determine a value of  $w_0$ , parameters  $\epsilon_a$  are fixed and the value with the smallest error rate of ear detection is selected. After training,  $w_0 = 0.5$  are obtained. For  $\epsilon_a$  the smaller the value we set, the higher accuracy

of the detection we can get, but the longer searching time is needed. In our experiments, we set  $\epsilon_a = -2.5$ .

We chose to test our algorithm in the PHP database because the PHP database includes most of variations in head pose. As most of the existing ear databases are taken under controlled conditions, we create an ear database named ‘‘WildEar’’, which includes 200 images captured from real world under uncontrolled conditions or collected from the Internet. All images in WildEar database are photographed with varying poses, different lighting and complicated background. For all the test images considered for the experiment, ground truth ear position is obtained by manually labeling each image prior to the experiment. As all the test images considered for this experiment contain true ears, the performance in terms of accuracy is described as:

$$Accuracy = \frac{\text{Number of true ear detection}}{\text{number of test images}} \times 100\% \quad (14)$$

In our experiments, if detected ear regions overlapping with ground-truth position is more than 50%, it is classified as successful detection. We compare the proposed method with the MHD based ear detection method [2], which is also based on the ear edge model. As the method in [2] is not invariant to affine transform, we also implement an affine invariant MHD based ear detection using our ETSS. Table 1 exhibits results of our proposed method and the other two approaches. We can see that the detection accuracy of the method in [2] is very low comparing to the other two approaches. That is because ear images in WildEar database have varying poses, and the MHD method in [2] is not invariant to rotation (in plane and out of plane). Our approach also performs better than affine invariant MHD with ETSS. The reason is that our approach incorporates structural information of line orientation and line-point association.

**Table 1.** The comparisons of our method with the other two state-of-the-art methods

Dataset	Methods	Ear detection accuracy (%)
WildEar	MHD [2]	43.50
	MHD with ETSS	87.50
	Our method	94.50
PHP dataset	EHT [8]	89.88
	Our method	92.35

We also compare our method with Entropy-cum-Hough-transform (EHT) based ear detection approach in [8], since EHT also has been evaluated using PHP Dataset. We selected all 93 pose-variant images from each person in PHP Dataset whose ears were not occluded. Thus, a total of 837 images from 9 subjects form this customized Head Pose database. It must be noted that authors of [8] only selected a total of 168 images without any occlusions from 12 subjects to form their customized Head Pose database. It shows that the proposed approach is able to outperform the state-of-the-art approach in [8].

Figure 3 shows some ear detection results using our method. The ear edge template was transformed and drawn on the test images using the located affine transform matrix. The top 2 rows provide examples of detection results with the varying pose, lighting conditions (indoor and outdoor) and extremely complicated background. We also tested the proposed technique on images taken from top to bottom and taken from bottom to top, as illustrated in third row of Fig. 3. This is one of the most likely situations in the practical application. The bottom row is the ear detection results in the images gathered from the web. Our results indicate that the proposed affine invariant ear detection method is a viable option for ear detection in the wild.



**Fig. 3.** Ear detection in the wild.

## 5 Conclusion

In this paper, we present a novel ear detection method under unconstrained setting based on the fast line segment Hausdorff distance and branch-and-bound scheme. The main contributions of this paper are twofold: (1) the proposed FLHD not only incorporates structural and spatial information to compute the similarity, but also needs less storage space and is faster than points based MHD. (2) A fast global search based on branch-and-bound scheme makes our method capable of handling arbitrary 2D affine transformations. Experiments showed that our approach can detect ears in the wild with varying pose and extremely complex background. Our method also can be used in affine invariant general planer object detection.

## References

1. Pei, S.-C., Liou, L.-G.: Finding the motion, position and orientation of a planar patch in 3D space from scaled-orthographic projection. *Pattern Recogn.* **27**(1), 9–25 (1994)
2. Sarangi, P.P., Panda, M., Mishra, B.S.P., Dehuri, S.: An automated ear localization technique based on modified hausdorff distance. In: Raman, B., Kumar, S., Roy, P.P., Sen, D. (eds.) *Proceedings of International Conference on Computer Vision and Image Processing*. AISC, vol. 460, pp. 229–240. Springer, Singapore (2017). [https://doi.org/10.1007/978-981-10-2107-7\\_21](https://doi.org/10.1007/978-981-10-2107-7_21)
3. Gao, Y., Leung, M.K.H.: Line segment Hausdorff distance on face matching. *Pattern Recogn.* **35**(2), 361–371 (2002)
4. Burge, M., Burger, W.: Ear biometrics in computer vision. In: *Proceedings 15th International Conference on Pattern Recognition*, pp. 822–826. IEEE, Barcelona (2000)
5. Hurley, D.J., Nixon, M.S., Carter, J.N.: Force field feature extraction for ear biometrics. *Comput. Vis. Image Understand.* **98**(3), 491–512 (2005)
6. Prakash, S., Jayaraman, U., Gupta, P.: Connected component based technique for automatic ear detection. In: *16<sup>th</sup> International Conference on Image Processing (ICIP)*, pp. 2741–2744. IEEE, USA (2009)
7. Pflug, A., Winterstein, A., Busch, C.: Robust localization of ears by feature level fusion and context information. In: *International Conference on Biometrics (ICB)*, pp. 1–8. IEEE, Madrid (2013)
8. Chidananda, P., Srinivas, P., Manikantan, K., Ramachandran, S.: Entropy-cum-hough-transform-based ear detection using ellipsoid particle swarm optimization. *Mach. Vis. Appl.* **26**(2), 185–203 (2015)
9. Emeršič, Ž., Gabriel, L.L., Štruc, V., Peer, P.: Pixel-wise ear detection with convolutional encoder-decoder networks. *arXiv* (2017)
10. Zhang, Y., Mu, Z.: Ear detection under uncontrolled conditions with multiple scale faster region-based convolutional neural networks. *Symmetry* **9**(4), 53 (2017)
11. Huttenlocher, D.P., Rucklidge, W.J., Klanderman, G.A.: Comparing images using the Hausdorff distance under translation. *IEEE Trans. Pattern Anal. Mach. Intell.* **15**(9), 654–656 (1993)
12. Dubuisson, M.-P., Jain, A.K.: A modified Hausdorff distance for object matching. In: *International Conference on Pattern Recognition*, pp. 566–568. IEEE, Jerusalem (1994)
13. Liu, M.-Y., Tuzel, O., Veeraraghavan, A., Chellappa, R.: Fast directional chamfer matching. In: *Computer Vision and Pattern Recognition (CVPR)*, pp. 1696–1703, IEEE, San Francisco (2010)
14. Kovese, P.D.: *MATLAB and octave functions for computer vision and image processing* (2008)
15. Fischer, J., Heun, V.: Space-efficient preprocessing schemes for range minimum queries on static arrays. *SIAM J. Comput.* **40**(2), 465–492 (2011)
16. Gourier, N., Hall, D., Crowley, J.L.: Estimating face orientation from robust detection of salient facial structures. In: *FG Net Workshop on Visual Observation of Deictic Gestures*, Cambridge, UK, pp. 17–25 (2004)
17. Gao, Y., Leung, M.: Face recognition using line edge map. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(6), 764–779 (2002)



# Line Voronoi Diagrams Using Elliptical Distances

Aysylu Gabdulkhakova<sup>(✉)</sup>, Maximilian Langer, Bernhard W. Langer,  
and Walter G. Kropatsch

Pattern Recognition and Image Processing Group,  
193-03 Institute of Visual Computing and Human-Centered Technology,  
Technische Universität Wien, Favoritenstrasse 9-11, Vienna, Austria  
{aysylu,mlanger,krw}@prip.tuwien.ac.at

**Abstract.** The paper introduces an Elliptical Line Voronoi diagram. In contrast to the classical approaches, it represents the line segment by its end points, and computes the distance from point to line segment using the Confocal Ellipse-based Distance. The proposed representation offers specific mathematical properties, prioritizes the sites of the greater length and corners with the obtuse angles without using an additional weighting scheme. The above characteristics are suitable for the practical applications such as skeletonization and shape smoothing.

**Keywords:** Confocal ellipses · Line Voronoi diagram  
Hausdorff distance

## 1 Introduction

Various branches of computer science - for example, pattern recognition, computer graphics, computer-aided design - deal with the problems that are inherently geometrical. In particular, Voronoi diagram is a fundamental geometrical construct that is successfully used in a wide range of computer vision applications (e.g. motion planning, skeletonization, clustering, and object recognition) [1]. It reflects the proximity of the points in space to the given site set.

On one side, proximity depends on a selected distance function. Existing approaches in  $\mathbb{R}^2$  explore the properties and application areas of particular metrics:  $L_1$  [2],  $L_2$  [3, 4],  $L_p$  [5]. Chew et al. [6] present the Voronoi diagrams for the convex distance functions. Klein et al. [7] introduced a concept of defining the properties of the Voronoi diagram for the classes of metrics, rather than analyzing each metric separately. A group of approaches proposes the site-specific weights, e.g. skew distance [8], power distance [9], crystal growth [10], and convex polygon-offset distance function [11]. This paper presents a new type of a Line

---

A. Gabdulkhakova—Supported by the Austrian Agency for International Cooperation in Education and Research (OeAD) within the OeAD Sonderstipendien program, and by the Faculty of Informatics.

Voronoi diagram that uses Confocal Ellipse-based Distance (CED) [12] as a metric of proximity. In contrast to Hausdorff Distance (HD), CED (1) defines the line segment by its two end points, (2) represents the propagation of the distance values from the line segment to the points in  $\mathbb{R}^2$  as confocal ellipses. The proposed geometrical construct reconsiders the classical Euclidean distance-based space tessellation, and introduces hyperbolic and elliptical cells, that have surprising mathematical properties. Structure is added to a set of points by putting the subsets of points in relation. The simplest relation that every structure should have is a binary relation relating two points. That is why a new metric relating points with pairs of points is extremely relevant for the community.

On the other side, proximity depends on the type of objects in the site set. Polygonal approximations of objects are commonly agreed to be used in a majority of geometric scenarios [13]. Therefore, in this paper the site set contains points and/or line segments.

The remainder of the paper is organized as follows. Section 2 presents the Elliptical Line Voronoi diagram (ELVD), provides an analysis of the proximity as defined by CED and HD, and introduces the Hausdorff ellipses. Section 3 shows the properties of ELVD with regard to the type of objects in the site set. Section 4 discusses the advantages of applying the ELVD to skeletonization and contour smoothing. Finally, the paper is concluded in Sect. 5.

## 2 Elliptical Line Voronoi Diagram (ELVD)

A Voronoi diagram partitions the Euclidean plane into *Voronoi cells* that are connected regions, where each point of the plane is closer to one of the given *sites* inside the cell. In the classical case the sites are a finite set of points and the metric used is the Euclidean distance.

In our contribution we extend the original definition by (1) considering a site to be a straight line segment, (2) measuring the proximity of a point to the site using the parameters of a unique ellipse that passes through this point and takes the two end points of the line segment as its focal points. We call the resultant geometrical construct Elliptical Line Voronoi diagram, or in short ELVD.

As opposed to Euclidean distance in Voronoi diagram, proximity in the ELVD is defined with respect to the Confocal Ellipse-based Distance. Similarly to the Blum's medial axis [14], ELVD can be extracted from the Confocal Elliptical Field (CEF) [12] as a set of points which have identical distance value for at least two sites.

### 2.1 Confocal Ellipse-Based Distance (CED)

Let  $\delta(M, N) = \sqrt{(M - N)^2}$ ,  $M, N \in \mathbb{R}^2$ , be the Euclidean distance between the points  $M$  and  $N$ .

**Definition 1.** The ellipse,  $E(F_1, F_2; a)$ <sup>1</sup> is the locus of points on a plane, for which the sum of the distances to two given points  $F_1$  and  $F_2$  (called focal points) is constant:

$$\delta(M, F_1) + \delta(M, F_2) = 2a, \tag{1}$$

where parameter  $a$  is the length of the semi-major axis of the ellipse.

Ellipses that have the same focal points  $F_1$  and  $F_2$  are called confocal ellipses. Given two focal points  $F_1$  and  $F_2$ , a family of confocal ellipses covers the whole plane. Each ellipse in this family is defined as  $E(a) = \{P \in \mathbb{R}^2 \mid \delta(P, F_1) + \delta(P, F_2) = 2a\}$ ,  $a \geq f$ . Here  $f = \frac{\delta(F_1, F_2)}{2}$  denotes half the distance between the two focal points  $F_1$  and  $F_2$ .

**Definition 2.** Let us consider two confocal ellipses  $E(a_1)$  and  $E(a_2)$  generated by focal points  $F_1, F_2 \in \mathbb{R}^2$ , where  $a_1, a_2 \geq f$ . The Confocal Ellipse-based Distance (CED) between  $E(a_1)$  and  $E(a_2)$ ,  $e: \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}$ , is determined as the absolute difference between the lengths of their major axes:

$$e(E(a_1), E(a_2)) = 2|a_1 - a_2| \tag{2}$$

CED is a metric and  $E(a_1) \subset E(a_2)$ , if  $a_1 < a_2$ .

## 2.2 Confocal Elliptical Field (CEF)

Consider a set of sites that contains the pairs of points:  $S = \{(F_1, F_2), (F_3, F_4), \dots, (F_{N-1}, F_N)\}$ . A site  $s = (F_i, F_{i+1}), i \in [1, \dots, N - 1]$  generates a family of confocal ellipses with  $F_i$  and  $F_{i+1}$  taken as the focal points. The distance from the point  $P \in \mathbb{R}^2$  to the site  $s$ , is defined with respect to CED as:

$$d(P, s) = e(E(a_P), E(a_0)) \tag{3}$$

where  $E(a_P)$  corresponds to the unique ellipse with focal points  $F_i$  and  $F_{i+1}$  that contains  $P$ ;  $E(a_0)$  corresponds to the ellipse with the same foci  $F_i$  and  $F_{i+1}$ , whose eccentricity equals 1. In other words, this distance is defined as:  $d(P, s) = \delta(P, F_i) + \delta(P, F_{i+1}) - \delta(F_i, F_{i+1}) = 2(a - f)$ .

**Definition 3.** Confocal Elliptical Field (CEF) is an operator that assigns to each point  $P \in \mathbb{R}^2$  its distance to the closest site from  $S$ :

$$CEF = d(P, S) = \inf\{d(P, s) \mid s \in S\} \tag{4}$$

**Definition 4.** Separating curve is a set of points in CEF that have an identical value as generated from multiple (more than one) distinct sites.

For the given set of sites that contain points and line segments, separating curves define the ELVD.

---

<sup>1</sup> If for several ellipses the focal points are the same, we denote it as  $E(a)$ .



### 2.3 Relation Between CED and Hausdorff Distance

As opposed to CEF, in classical Line Voronoi diagram, the line segment is a set of all points that form it. Therefore, for each point in space the proximity to the line segment can be defined with respect to the Hausdorff Distance.

**Definition 5.** *The Hausdorff Distance (HD) between a point  $P$  and a set of points  $T$  is defined as the minimum distance of  $P$  to any point in  $T$ . Usually the distance is considered to be Euclidean:*

$$HD = d_H(P, T) = \inf\{\delta(P, t) \mid t \in T\} \tag{5}$$

By introducing a scaling factor of  $\frac{1}{2}$  for the CED we obtain the same distance field for HD and CED, in case the two focal points coincide. Another property is that the  $\lambda$ -isoline of the CED  $\{P \mid d(P, s) = \lambda\}$  encloses the  $r$ -isoline of HD  $\{P \mid d_H(P, T) = r\}$ , with  $s$  being a site containing the two foci  $F_1$  and  $F_2$ ,  $T$  is a set of points that form the line segment  $\overline{F_1 F_2}$ . Figure 1a shows multiple isolines for HD and CED that have the same  $\lambda$  and  $r$ . Note that both, HD and CED, have zero distance values along the line segment  $\overline{F_1 F_2}$ .

We can derive a value  $\lambda$  for any given  $r$  so that the CED  $\lambda$ -isoline is enclosed by the HD  $r$ -isoline (see Fig. 1b). To find  $\lambda$  we are looking for the value where the minor ellipse radius  $b$  equals  $r$ . In an ellipse  $b^2 = a^2 - f^2$ , that in this case can be reformulated to  $r^2 = a^2 - f^2$ , solving for  $a$ :

$$\lambda = 2a - f = 2\sqrt{r^2 + f^2} - f. \tag{6}$$

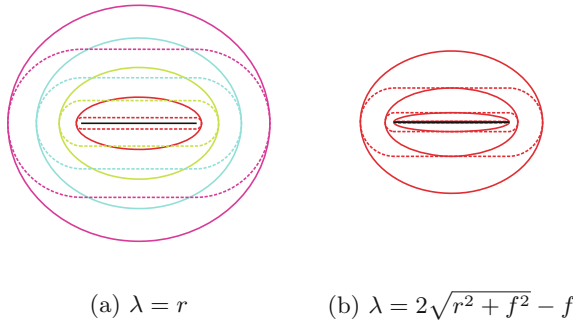
By similar reasoning we can also derive  $r$  for a given  $\lambda$  that will ensure the  $r$ -isoline of the HD is enclosed by the CED  $\lambda$ -isoline:

$$r = \sqrt{2f\lambda + \lambda^2}. \tag{7}$$

We can construct ellipses around a line segment by starting with a distance  $\lambda_0 = 1$  and increasing according to the sequence:

$$\lambda_{n+1} = \sqrt{2f\lambda_n + \lambda_n^2} \tag{8}$$

We name these isolines *Hausdorff Ellipses* of a line segment.



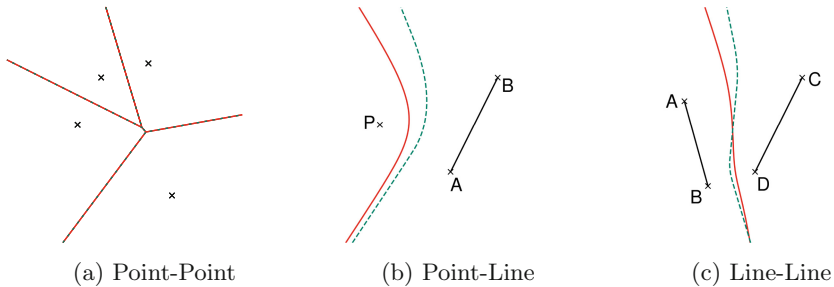
**Fig. 1.** Comparison of HD (dashed) and CED (solid) isolines

### 3 Properties of ELVD

The proximity depends not only on the type of metric used, but also on the type of object in the site set. In this paper site is considered to be a point or a line segment. According to the Definition 3 of CEF, the distance field of a point contains concentric circles, and of a line segment - confocal ellipses. Thus, the separating curve varies according to the different combinations of the site types.

#### 3.1 Point and Point

In terms of CED, the site that represents a point contains identical foci. The resultant distance field of each site is formed by concentric circles. The separating curves are the perpendicular bisectors, and the ELVD is identical to the Voronoi diagram with Euclidean distance (Fig. 2a).



**Fig. 2.** Comparison of ELVD (solid red) and Voronoi diagram (dashed green). (Color figure online)

#### 3.2 Point and Line

Consider the site set that contains point  $P$  and line segment  $(A, B)$ . The receptive field of the point  $P$  depends on the position of the line segment, and ELVD is represented by a higher-order curve (Fig. 2b).

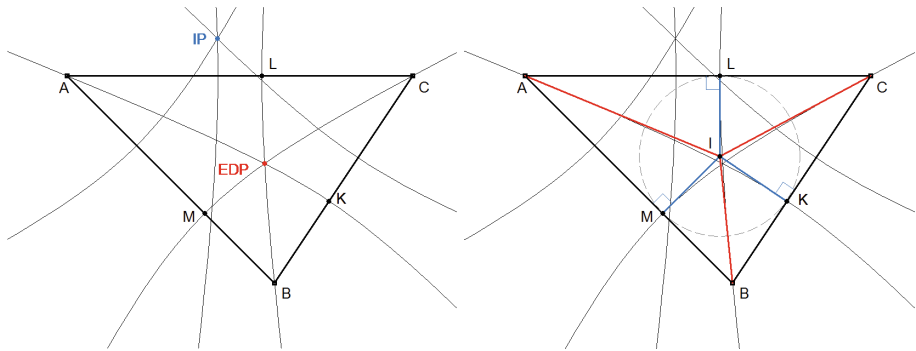
#### 3.3 Line and Line

For the site set that contains two line segments  $(A, B)$  and  $(C, D)$ , the ELVD is represented by a high-order curve of a different nature than for the Point-Line case (see Fig. 2c). The steepness and the shape of the curve depends on the length of the line segments, and their mutual arrangement (parallel, intersecting, non-intersecting). The mutual arrangement does not consider  $(A, B)$  and  $(C, D)$  to be connected as a polygon, i.e.  $B \neq C$ . This case is covered in Sect. 3.5.

### 3.4 Triangle

The simplest closed polygonal shape - a triangle - can be represented by:

- *three points corresponding to its vertices*  
 In the classical Voronoi diagram on the point set, the separation curves of the (Delaunay-) triangle are the perpendicular bisectors of its edges, they intersect at the center of the circumscribed circle.
- *by a set of  $N$  points, that form the contour of the triangle*  
 In the extension of the classical Line Voronoi diagram on the line set using the Euclidean distance, the separating curves of the triangle are its angular bisectors which intersect at the center of the incircle.
- *by three line segments corresponding to the edges of the triangle*  
 For the ELVD the separating curve between the two line segments that share one endpoint is a hyperbolic branch [12]. Therefore, the separation curves in the triangle are three hyperbolic branches, each passing through one vertex of the triangle, i.e.  $A, B$  or  $C$ , and intersecting the sides at the points  $K, L, M$  respectively (Fig. 3a).



(a) Hyperbolic branches of the ELVD in- (b) The tangents on the hyperbola in the  
 tersect at the Equal Detour Point (EDP) intersection points A, B, C and K, L, M  
 and Isoperimetric Point (IP). intersect at the incenter (I).

**Fig. 3.** Properties of the Equal Detour Point, Isoperimetric Point and incenter.

The separating curves of the triangle as obtained from ELVD have the following geometric properties:

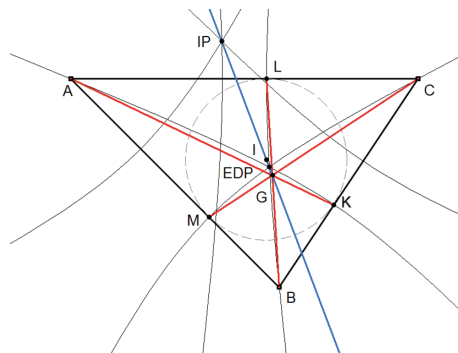
1. The separating curves intersect at a common point, known in the literature as the *Equal Detour Point (EDP)* [15] (see Fig. 3a).
2. The complementary branches of the hyperbolas intersect at a common point, known as the *Isoperimetric Point (IP)* [15] (Fig. 3a).
3. The six tangents of the hyperbolas at the six points  $A, B, C$ , and  $K, L, M$  intersect all at the center of the incircle  $I$  (Fig. 3b).

4. The intersection  $EDP$  of the three hyperbolas is located inside the triangle formed by the shortest side of the triangle and  $I$  (Fig. 3b).
5. The tangents at the triangle's corners  $A, B, C$  are the angular bisectors of the two adjacent sides respectively (Fig. 3b).
6. The three tangents at  $K, L, M$  form a right angle while intersecting the edges of the triangle (Fig. 3b).
7. The hyperbola chords  $AK, BL$  and  $IM$  intersect at the *Gergonne point* ( $G$ ) [15] (Fig. 4).
8. The  $EDP$  distance value of the CEF equals the radius of the inner Soddy circle.

Let  $P \in \mathbb{R}^2$  be an  $EDP$ , and  $K, L, M$  - be the points of intersection between separating curves and the edges of the triangle  $\triangle ABC$ . Consider the following distances: (1)  $r_P = CEF(P)$  - distance value at  $P$  in the confocal elliptical field; (2)  $r_A = \delta(A, M) = \delta(A, L)$ ; (3)  $r_B = \delta(B, M) = \delta(B, K)$ ; (4)  $r_C = \delta(C, L) = \delta(C, K)$ . The circle with the center at  $P$  and radius  $r_P$  is an inner Soddy circle [16], thus, it is tangent to the circles with the centers at  $A, B, C$  and radii  $r_A, r_B, r_C$  correspondingly. This property is valid not only for the  $EDP$ , but for all points of the separation hyperbola branches that lie on the curves  $PM, PK$ , and  $PL$ . In addition, according to the Soddy theorem, the following equation holds true:

$$\left(\frac{1}{r_A} + \frac{1}{r_B} + \frac{1}{r_C} + \frac{1}{r_P}\right)^2 = 2\left(\frac{1}{r_A^2} + \frac{1}{r_B^2} + \frac{1}{r_C^2} + \frac{1}{r_P^2}\right) \tag{9}$$

In case of a regular triangle, radii  $r_A, r_B, r_C$  are identical. Otherwise, their values vary depending on the angle at the corresponding vertex, and length of the edges that contain this vertex. The ELVD implicitly encodes the weighting factors, as compared to the classical Voronoi diagram.



**Fig. 4.** The incenter ( $I$ ), Gergonne point ( $G$ ), Isoperimetric Point ( $IP$ ) and Equal Detour Point ( $EDP$ ) are collinear.

### 3.5 Polygon

Consider a site set that defines an open polygon  $S = \{(F_1, F_2), \dots, (F_{N-1}, F_N)\}$ ,  $N \in \mathbb{R}$ . For any  $s_i = (F_i, F_{i+1})$ ,  $F_i \neq F_{i+1}$ ,  $s_i \in S$ ,  $i \in [1, N - 1]$ .

If the sites are consecutive, i.e. have a common point  $F_i$ , the separating curve is a branch of a hyperbola that passes through  $F_i$ ,  $i \in [1, N]$  [12]. If the sites are non-consecutive, but their receptive fields overlap (e.g. the sites cross each other), then the separating curve is defined as in *Line and Line* case.

Let  $P$  be the point of intersection of two separating curves  $H_{F_i}$  and  $H_{F_{i+1}}$ , that pass through  $F_i$  and  $F_{i+1}$  correspondingly. For the triangle  $\Delta F_i P F_{i+1}$  the separation hyperbola branch that passes through  $P$  and intersects  $(F_i, F_{i+1})$  at the point  $M$  defines the following distances:  $r_{F_i} = \delta(F_i, M)$ ,  $r_{F_{i+1}} = \delta(F_{i+1}, M)$ . The circle with the center at  $P$  and radius  $r_P$  is tangent to the circles with centers at  $F_i, F_{i+1}$  and radii  $r_{F_i}, r_{F_{i+1}}$  respectively. This property holds true for all points on the separating curve between  $P$  and  $M$ .

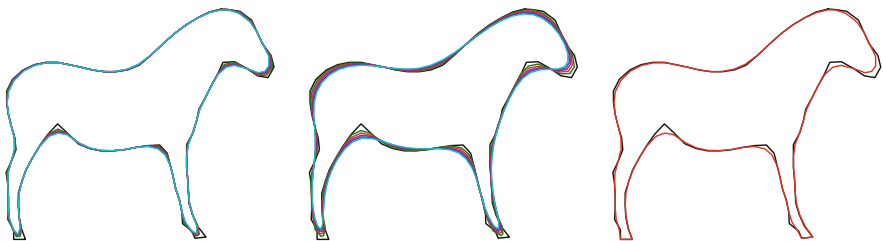
## 4 Applications

In this section we discuss the properties of ELVD that are valuable for the practical problems on an example of contour smoothing and skeletonization.

### 4.1 Contour Smoothing

By considering three successive points  $P_{i-1}$ ,  $P_i$  and  $P_{i+1}$  on a contour as a triangle  $\Delta_i$  we can smooth the contour by replacing the middle point  $P_i$  with the *EDP* of the triangle  $\Delta_i$ . Conventional average smoothing is related to the centroid of the triangle  $\Delta_i$ . This smoothing procedure can be iteratively repeated. Figure 5 shows a comparison between *EDP*-based smoothing and Mean-based smoothing, i.e. averaging over three successive contour points. Note that *EDP*-based smoothing does not affect low frequencies as much as high frequencies.

Let us denote the angles in the triangle  $\Delta_i$  as  $\alpha, \beta, \gamma$ . The angles formed by the vertices of the triangle and the incenter are  $\frac{\pi+\alpha}{2}, \frac{\pi+\beta}{2}, \frac{\pi+\gamma}{2}$ . This means



(a) *EDP*-based smoothing (b) Mean-based smoothing (c) Preserved sharp corners

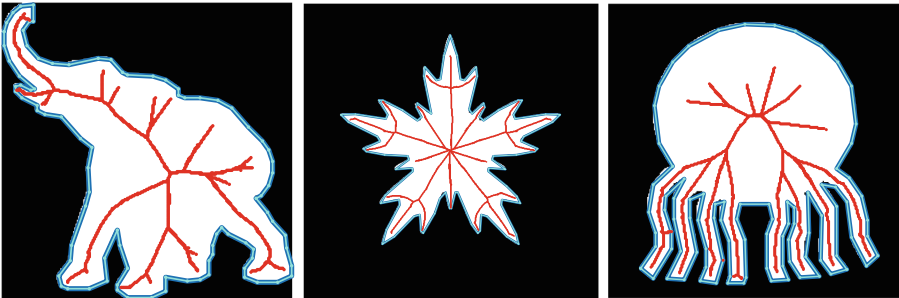
**Fig. 5.** Contour smoothing achieved by five iterations.

that the sharp angle ( $< \frac{\pi}{2}$ ) will be replaced by the obtuse angle after smoothing. The shortest side has the smallest opposite angle and an angle of more than  $\frac{\pi}{2}$  is always the largest in a triangle. Hence: (1) the shortest side before smoothing becomes the longest, (2) the smoothing slows down with more iterations. According to the ELVD Properties 4 and 8, in case of a triangle, the same holds true for the *EDP*. The difference is that the incenter is equidistant from the corner sides, whereas *EDP* is closer to the shorter edge and obtuser angle than the incenter. This property is important in case of the outliers - the contour is smoothed with the less number of iterations.

Additionally we can preserve selected sharp corners by including the same point twice in the contour. Figure 5c gives an example of preserved sharp corners in the hooves of the horse.

## 4.2 Skeletonization

The ELVD can be successfully applied to create a skeleton of the shape [12], where the weighting is implicitly encoded in the length of the site (see Fig. 6). As compared to the classical Voronoi diagram-based skeletonization, the sites contain pairs of vertices. The skeletal points are not equidistant from the opposite sides of the shape - they are shifted towards the sites that represent the shorter edges. As a result, the longer edges have a greater receptive field.



**Fig. 6.** Examples of the ELVD-based skeletons (red). The polygonal approximation of the shape (cyan) contains 90 vertices in each case. (Color figure online)

## 5 Conclusion and Outlook

This paper presents a novel approach to the line Voronoi diagram by considering the distance from the point to the line segment by CED. The discussion of the ELVD proximity (from the point of metric and types of objects in the site set) shows that the classical Voronoi diagram is a special case of ELVD. The proposed approach has also the practical value: (1) skeletonization algorithm enables prioritization of the longer edges without extra weighting schema, (2) smoothing

of the shape enables a closer approximation of the contour and preservation of the sharp corners. The ongoing research considers ELVD properties regarding the weighting factors and the semantic interpretation of the corresponding geometrical construct.

## References

1. Aurenhammer, F.: Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Comput. Surv. (CSUR)* **23**(3), 345–405 (1991)
2. Hwang, F.K.: An  $O(n \log n)$  algorithm for rectilinear minimal spanning trees. *J. ACM (JACM)* **26**(2), 177–182 (1979)
3. Fortune, S.J.: A fast algorithm for polygon containment by translation. In: Brauer, W. (ed.) *ICALP 1985. LNCS*, vol. 194, pp. 189–198. Springer, Heidelberg (1985). <https://doi.org/10.1007/BFb0015744>
4. Edelsbrunner, H.: *Algorithms in Combinatorial Geometry*. EATCS Monographs on Theoretical Computer Science. Springer, Heidelberg (1987). <https://doi.org/10.1007/978-3-642-61568-9>
5. Lee, D.-T.: Two-dimensional Voronoi diagrams in the  $L_p$ -metric. *J. ACM (JACM)* **27**(4), 604–618 (1980)
6. Chew, L.P., Dyrsdale III, R.L.S.: Voronoi diagrams based on convex distance functions. In: *Proceedings of the First Annual Symposium on Computational Geometry*, pp. 235–244 (1985)
7. Klein, R., Wood, D.: Voronoi diagrams based on general metrics in the plane. In: Cori, R., Wirsing, M. (eds.) *STACS 1988. LNCS*, vol. 294, pp. 281–291. Springer, Heidelberg (1988). <https://doi.org/10.1007/BFb0035852>
8. Aichholzer, O., Aurenhammer, F., Chen, D.Z., Lee, D., Papadopoulou, E.: Skew Voronoi diagrams. *Int. J. Comput. Geom. Appl.* **9**(03), 235–247 (1999)
9. Aurenhammer, F.: Power diagrams: properties, algorithms and applications. *SIAM J. Comput.* **16**(1), 78–96 (1987)
10. Schaudt, B.F., Drysdale, R.L.: Multiplicatively weighted crystal growth Voronoi diagrams. In: *Proceedings of the Seventh Annual Symposium on Computational Geometry*, pp. 214–223. ACM (1991)
11. Barequet, G., Dickerson, M.T., Goodrich, M.T.: Voronoi diagrams for convex polygon-offset distance functions. *Discrete Comput. Geom.* **25**(2), 271–291 (2001)
12. Gabdulkhakova, A., Kropatsch, W.G.: Confocal ellipse-based distance and confocal elliptical field for polygonal shapes. In: *Proceedings of the 24th International Conference on Pattern Recognition, ICPR* (in print)
13. Aurenhammer, F., Klein, R., Lee, D.-T.: *Voronoi Diagrams and Delaunay Triangulations*. World Scientific Publishing Company, Singapore (2013)
14. Blum, H.: A transformation for extracting new descriptors of shape. In: *Models for Perception of Speech and Visual Forms*, pp. 362–380 (1967)
15. Veldkamp, G.R.: The isoperimetric point and the point(s) of equal detour in a triangle. *Am. Math. Mon.* **92**(8), 546–558 (1985)
16. Soddy, F.: The Kiss precise. *Nature* **137**, 1021 (1936)

# **Structural Matching**





# Modelling the Generalised Median Correspondence Through an Edit Distance

Carlos Francisco Moreno-García<sup>1</sup> and Francesc Serratosa<sup>2</sup>(✉)

<sup>1</sup> The Robert Gordon University, Garthdee Road, Aberdeen, Scotland, UK

<sup>2</sup> Universitat Rovira i Virgili, Av. Paisos Catalans 26, Tarragona, Catalonia, Spain  
`francesc.serratosa@urv.cat`

**Abstract.** On the one hand, classification applications modelled by structural pattern recognition, in which elements are represented as strings, trees or graphs, have been used for the last thirty years. In these models, structural distances are modelled as the correspondence (also called matching or labelling) between all the local elements (for instance nodes or edges) that generates the minimum sum of local distances. On the other hand, the generalised median is a well-known concept used to obtain a reliable prototype of data such as strings, graphs and data clusters. Recently, the structural distance and the generalised median has been put together to define a generalise median of matchings to solve some classification and learning applications. In this paper, we present an improvement in which the Correspondence edit distance is used instead of the classical Hamming distance. Experimental validation shows that the new approach obtains better results in reasonable runtime compared to other median calculation strategies.

**Keywords:** Generalised median · Edit distance · Optimisation  
Weighted mean

## 1 Introduction

A correspondence is defined as the result of a bijective function which designates a set of one-to-one mappings between elements representing the local information of two structures i.e. sets of points, strings, trees, graphs or data clusters. Each element (a point for sets of points; a character for strings, or a node and its edges for trees or graphs) has a set of attributes that contain specific information. Correspondences are usually generated, either manually or automatically, with the purpose of finding the similarity or a distance between two structures. In the case that correspondences are deduced through an automatic method, this is most commonly done through an optimisation process called matching. Several matching methods have been proposed for set of points [32], strings [25], trees and graphs [29].

Correspondences are used in various frameworks such as measuring the accuracy of different graph matching algorithms [4, 31], improving the quality of other correspondences [5], learning edit costs for matching algorithms [6], estimating the pose of a fleet of robots [7], performing classification [17] or calculating the consensus of a set of correspondences [18–21]. While most of these methods use the classical Hamming distance (HD) to calculate the dissimilarity between a pair of correspondences, in [23] authors have shown that this distance does not always reflect the dissimilarity between a pair of correspondences, and thus, a new distance called Correspondence Edit Distance (CED) was defined.

The median of a set of structures is roughly defined as a sample that achieves the minimum sum of distances (SOD) to all members of such set. This concept has been largely considered as a suitable representative prototype of a set [13] because of its robustness. For the case of strings [3], graphs [2], and data clusters [11], computing the median is an *NP*-complete problem. Thus, some suboptimal methods have been presented to calculate an approximation to the median. For instance, an embedding approach has been presented for strings [14], graphs [8] and data clusters [10]. Likewise, a strategy known as the evolutionary method for strings [9] and correspondences [22] has proven to obtain fair approximations to the median in reasonable time. Moreover, [22] presented a minimisation method which obtains the median using optimisation functions based on the HD. This work proved that it is possible to obtain the exact median for a set of correspondences using this framework, provided that the distance considered between the correspondences is the HD. In this paper our work is devoted towards revisiting the median calculation frameworks presented in [22], this time using the CED.

The rest of the paper is structured as follows. Section 2 establishes the basic definitions. Afterwards, in Sect. 3 we present the method to calculate the generalised median based on the CED. Then, Sect. 4 provides an experimental validation of the method. Finally, Sect. 5 is reserved for the conclusions and further work.

## 2 Basic Definitions

### 2.1 Distance Between Structures

Consider a structure  $G = (\Sigma, \mu)$ , where  $v_i \in \Sigma$  denotes the elements (i.e. local information) and  $\mu$  is a function that assigns a set of attributes to each element. This structure may contain null elements which have a set of attributes that differentiate them from the rest. We refer onwards to these null elements of  $G$  as  $\hat{\Sigma} \subseteq \Sigma$ . Moreover, given  $G = (\Sigma, \mu)$  and  $G' = (\Sigma', \mu')$  of the same order  $n$  (naturally or due to the aforementioned null element presence), we define the set of all possible correspondences  $T$ , such that each correspondence in  $T$  maps all elements of  $G$  to elements of  $G'$ ,  $f : \Sigma \rightarrow \Sigma'$  in a bijective manner.

For structures such as strings [30], trees [1] and graphs [12, 26, 28], one of the most widely used frameworks to calculate the distance is the edit distance.

The edit distance is defined as the minimum amount of required operations that transform one object into the other. To this end, several distortions or edit operations, consisting of insertion, deletion and substitution of elements are defined. Edit cost functions are introduced to quantitatively evaluate the edit operations. The basic idea is to assign a penalty cost to each edit operation considering the amount of distortion that it introduces in the transformation. Substitutions simply indicate element-to-element mappings. Deletions are transformed to assignments of a non-null element of the first structure to a null element of the second structure. Insertions are transformed to assignments of a non-null element of the second structure to a null element of the first structure. Given  $G$  and  $G'$  and a correspondence  $f$  between them, the edit distance is obtained as follows:

$$EditCost(G, G', f) = \sum_{\substack{v_i \in \Sigma - \hat{\Sigma} \\ v'_j \in \Sigma' - \hat{\Sigma}'}} d(v_i, v'_j) + \sum_{\substack{v_i \in \Sigma - \hat{\Sigma} \\ v'_j \in \hat{\Sigma}'}} K + \sum_{\substack{v_i \in \hat{\Sigma} \\ v'_j \in \Sigma - \hat{\Sigma}}} K \quad (1)$$

where  $f(v_i) = v'_j$  and function  $d$  is a distance function between the mapped elements. Moreover,  $K$  is a penalty cost for the insertion and deletion of elements. Thus, the edit distance  $ED$  is defined as the minimum cost under any bijection in  $T$ :

$$ED(G, G') = \min_{f \in T} EditCost(G, G', f) \quad (2)$$

### 2.2 Mean, Weighted Mean and Median

In its most general form, the mean of two structures  $G$  and  $G'$  is defined as a structure  $\bar{G}$  such that:

$$Dist(G, \bar{G}) = Dist(\bar{G}, G') \quad \text{and} \quad Dist(G, G') = Dist(G, \bar{G}) + Dist(\bar{G}, G') \quad (3)$$

where  $Dist$  is any distance metric defined on the domain of these structures. Moreover, the concept of weighted mean is used to gauge the importance or the contribution of the involved structures in the mean calculation. The weighted mean between two structures is defined as:

$$Dist(G, \bar{G}) = \lambda \quad \text{and} \quad Dist(G, G') = \lambda + Dist(\bar{G}, G') \quad (4)$$

where  $\lambda$  is a constant that controls the contribution of the structures and holds  $0 \leq \lambda \leq Dist(G, G')$ .  $G$  and  $G'$  satisfy this condition, and therefore are also weighted means of themselves.

From the definition of the median, two different approaches are identified: the set median (SM) or the generalised median (GM). The first one is defined as the structure within the set which has the minimum SOD. Conversely, the GM is the structure out of any element in the set which obtains the minimum SOD.

### 2.3 Distance Between Correspondences

Given structures  $G$  and  $G'$  and two correspondences  $f^1$  and  $f^2$  between them, we proceed to define the HD and the CED.

**Hamming Distance.** The HD is defined as:

$$HD(f^1, f^2) = \sum_{i=1}^n (1 - \delta(v'_a, v'_b)) \tag{5}$$

where  $a$  and  $b$  such that  $f^1(v_i) = v'_a$  and  $f^2(v_i) = v'_b$ , and  $\delta$  being the Kronecker Delta function:

$$\delta(x, y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

**Correspondence Edit Distance.** The CED is defined, in a similar way to Eqs. 1 and 2, as:

$$CED(f^1, f^2) = \min_{h \in H} Corr\_EditCost(f^1, f^2, h) \tag{7}$$

where

$$Corr\_EditCost(f^1, f^2, h) = \sum_{\substack{m_i^1 \in M^1 - \hat{M}^1 \\ m_k^2 \in M^2 - \hat{M}^2}} d(m_i^1, m_k^2) + \sum_{\substack{m_i^1 \in M^1 - \hat{M}^1 \\ m_k^2 \in \hat{M}^2}} K + \sum_{\substack{m_i^1 \in \hat{M}^1 \\ m_k^2 \in M^2 - \hat{M}^2}} K \tag{8}$$

where  $M^1$  and  $M^2$  are the sets of all possible mappings,  $\hat{M}^1$  and  $\hat{M}^2$  are the sets of null mappings.

The distance between mappings,  $d(m_i^1, m_k^2)$  was defined using Eq. 9 as:

$$d(m_i^1, m_k^2) = dn(v_i, v_k) + dn(f^1(v_i), f^2(v_k)) \tag{9}$$

where  $dn$  is a distance between the local parts of the structures, which is application dependent.

Notice that the elements used by CED are the mappings within  $f^1$  and  $f^2$ . More formally, correspondences  $f^1$  and  $f^2$  are defined as sets of mappings  $f^1 = m_1^1, \dots, m_i^1, \dots, m_n^1$  and  $f^2 = m_1^2, \dots, m_k^2, \dots, m_n^2$ , where  $m_i^1 = (v_i, f^1(v_i))$  and  $m_k^2 = (v_k, f^2(v_k))$ .

### 2.4 Generalised Median Correspondence Based on the Hamming Distance

In [22], authors presented a method to calculate the exact GM  $\hat{f}$  of a set of correspondences based on the HD. Such method is based on converting a set of correspondences  $f^1, \dots, f^i, \dots, f^m$  into correspondence matrices  $F^1, \dots, F^i, \dots, F^m$ . Afterwards, a linear solver [15, 16, 24] is applied to the sum of these matrices as follows:

$$\hat{f} = \underset{i=1}{\operatorname{argmin}} \sum^n (C \circ F^i[x, y]) \tag{10}$$

where  $[x, y]$  is a specific cell and  $C$  is the following matrix:

$$C = \sum_{i=1}^m (\mathbf{1} - F^i[x, y]) \tag{11}$$

where

$$F^i[x, y] = \begin{cases} 1 & \text{if } f^i(v_x) = v'_y \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

The idea is that by introducing a value of either 0 or a 1 in the correspondence matrix, the HD is being considered and thus minimised by the method.

## 3 Methodology

The aim of this paper is to model the GM of a set of correspondences through the CED. As commented in the introduction, it only has been modelled through the HD and we supposed that through the CED, much more interesting or useful median could be generated from an application point of view. Therefore, we only want to redefine matrix  $C$  in Eq. 11 since the current one makes the median to be generated through the HD. Equation 13 shows our proposal:

$$C = \sum_{i=1}^n B^i[x, y] \tag{13}$$

where

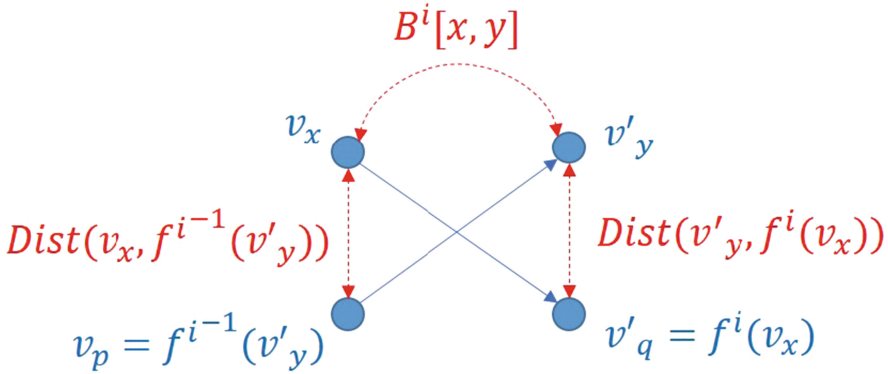
$$B^i[x, y] = \operatorname{Dist}(v_x, f^{i-1}(v'_y)) + \operatorname{Dist}(v'_y, f^i(v_x)) \tag{14}$$

Suppose that  $m$  is the mapping  $m = \{v_x, v'_y\}$ . Then,  $B^i[x, y]$  is defined as the distance between this supposed mapping  $f(v_x) = v'_y$  and the mappings imposed by correspondence  $f^i$  that relates elements  $v_x$  and  $v'_y$ . That is,

$$B^i[x, y] = d(m, m_x^i) + d(m, m_y^i) \tag{15}$$

As the distance between two mappings becomes higher, so does the value of  $B^i[x, y]$ . Likewise, the value of  $(\mathbf{1} - F^i[x, y])$  in Eq. 11 is higher for mappings that are not present in any correspondence of the set. As a result, matrix  $C$  in Eq. 13 is a generalisation of matrix  $C$  in Eq. 11.

Finally, considering Eqs. 9 and 15, we arrive to Eq. 14. Figure 1 graphically shows the computation of  $B^i[x, y]$ :



**Fig. 1.**  $\circ \rightarrow$ : Mappings in correspondences.  $\dashrightarrow$ : Computation of the distance

Notice that the first part of the expression is similar to how the bijective function  $h$  is calculated in Eq. 7, in the sense that it only computes the distance between mappings that have the same element on the output structure  $G$ . Moreover, notice that according to the  $Dist$  measure used, null elements (and thus null mappings) are considered accordingly. Finally, matrix  $C$  is minimised in the same way as in Eq. 10.

### 4 Validation

The experimental validation was carried out as follows. We have generated two repositories  $S^5$  (with graphs/correspondences of a cardinality of 5 nodes/mappings) and  $S^{30}$  (with graphs/correspondences of a cardinality of 30 nodes/mappings), with the attributes of the nodes being real numbers, and edges being unattributed and conformed through the Delaunay triangulation. Each repository is integrated by 3 datasets consisting of 60 8-tuples  $s_1 = \{G_1, G'_1, f_1^1, \dots, f_1^6\}, \dots, s_i = \{G_i, G'_i, f_i^1, \dots, f_i^6\}, \dots, s_{60} = \{G_{60}, G'_{60}, f_{60}^1, \dots, f_{60}^6\}$ . All correspondences for each dataset are obtained through the following three correspondence generation scenarios:

- Completely at random: Six bijective correspondences are randomly generated for each tuple.
- Evenly distributed: From a “seed” bijective correspondence generated using [27], two mappings are swapped randomly and a new correspondence is created. This process is repeated six times for each tuple. The seed correspondence is not included in the tuple.
- Unevenly distributed: From a “seed” bijective correspondence generated using [27], pairs of mappings are swapped a random number of times and a new correspondence is created. This process is repeated six times for each tuple. Due to the randomness of the swaps, the seed correspondence may be included in the tuple.

The median was calculated for HD and CED by using the following methods:

1. SM as the correspondence in the set with the lowest SOD (A\* method).
2. Evolutionary method for GM correspondence approximation presented in [22] (EVOL1).
3. Evolutionary method for GM correspondence approximation presented in [22] using a modified weighted mean search strategy (EVOL2).
4. Minimisation method (Min-GM). Method presented in [22] for HD and the method presented in this paper for CED.

Tables 1, 2 and 3 shows the average SOD of the mean with respect to the set ( $SOD_{AVG}$ ), the reduction percentage of SOD of methods 2, 3 and 4 with respect to 1 (RED) and the average runtime in seconds (RUN) for the three datasets in the two repositories. Notice that since the HD and the CED are distances which exist in different spaces, a comparison of  $SOD_{AVG}$  results between HD and CED methods is not viable. Moreover, RED scores are mostly meant to illustrate the improvement of each method with respect to the SM in its own distance space, since the increment of HD is linear while CED depends on the attributes of the graphs.

For the “Completely at random” datasets, Table 1 shows lower  $SOD_{AVG}$  values for Min-GM than for the rest of methods on both  $S^5$  and  $S^{30}$ . Moreover, it can be observed that Min-GM achieves a 10% RED on the dataset in the  $S^{30}$  repository. However, this case is also the one that takes the most time to be computed. In contrast, although RED is not that considerable for Min-GM in the HD case, the runtime for this method is always comparable to the SM calculation. Finally, it can be noticed that EVOL1 never outperforms the SM, while EVOL2 does for the dataset in  $S^{30}$ . Both EVOL1 and EVOL2 have similar runtimes.

**Table 1.** Average SOD ( $SOD_{AVG}$ ), reduction percentage of average SOD with respect to SM (RED) and runtime (RUN) using the “Completely at random” scenario.

		Completely at random					
		$S^5$			$S^{30}$		
		$SOD_{AVG}$	RED	RUN	$SOD_{AVG}$	RED	RUN
HD	SM	19	-	0.0009	141	-	0.01
	MIN-GM	18	6	0.002	137	3	0.008
	EVOL1	19	0	0.004	141	0	0.1
	EVOL2	19	0	0.009	139	1.5	0.2
CED	SM	62000	-	0.01	642000	-	4.4
	MIN-GM	60000	4	0.02	580000	10	9.3
	EVOL1	62000	0	0.014	642000	0	4.7
	EVOL2	62000	0	0.007	628000	3	4.8

In the “Evenly distributed” datasets shown in Table 2, the best  $SOD_{AVG}$  and RED results are obtained by Min-GM. In fact, this experiment proves that Min-GM always obtains the exact GM, given that the median calculated for  $S^5$  and  $S^{30}$  always has a SOD of 12 towards the correspondences in the set. This value results from multiplying the number of correspondences (six) times the mappings swapped from the seed correspondence (two), which is known in advance to be the GM. Given the attribute dependant nature of the CED, this rule is not visible for the  $SOD_{AVG}$  and thus RED scores of Min-GM using CED appear to be lower compared to Min-GM using HD.

**Table 2.** Average SOD ( $SOD_{AVG}$ ), reduction percentage of average SOD with respect to SM (RED) and runtime (RUN) using the “Evenly distributed” scenario.

		Evenly distributed					
		$S^5$			$S^{30}$		
		$SOD_{AVG}$	RED	RUN	$SOD_{AVG}$	RED	RUN
HD	SM	13	-	0.006	19	-	0.01
	Min-GM	12	8	0.002	12	37	0.003
	EVOL1	13	0	0.003	15	22	0.004
	EVOL2	13	0	0.007	14	27	0.02
CED	SM	18400	-	0.02	63100	-	4.1
	Min-GM	18100	2	0.03	49300	22	9
	EVOL1	18400	0	0.003	63100	0	3.5
	EVOL2	18400	0	0.007	59000	7	3.5

**Table 3.** Average SOD ( $SOD_{AVG}$ ), reduction percentage of average SOD with respect to SM (RED) and runtime (RUN) using the “Unevenly distributed” scenario.

		Unevenly distributed					
		$S^5$			$S^{30}$		
		$SOD_{AVG}$	RED	RUN	$SOD_{AVG}$	RED	RUN
HD	SM	17	-	0.006	66	-	0.001
	MIN-GM	16	6	0.002	53	20	0.003
	EVOL1	17	0	0.003	65	22	0.006
	EVOL2	17	0	0.007	64	27	0.02
CED	SM	76500	-	0.005	839000	-	4.9
	MIN-GM	69100	10	0.002	669000	21	9.9
	EVOL1	76500	0	0.006	839000	0	5.3
	EVOL2	765000	0	0.01	779000	8	5.3

Finally, Table 3 shows the results for the “Unevenly distributed” datasets, where although the GM may be included in the set, larger  $SOD_{AVG}$  values are



obtained compared to the previous two scenarios. In this case, it is observed that RED is larger for Min-GM using CED than for HD. Nonetheless, the computation of Min-GM using CED for the  $S^{30}$  dataset conveys the largest runtime. Meanwhile, EVOL1 and EVOL2 maintain a similar trend to the previous two scenarios.

The following conclusions can be drawn from these experiments. If the correspondences have a low number of mappings or high precision is required, then Min-GM with CED is the best option. In contrast, HD has a better accuracy to runtime trade-off for correspondences with a high mapping order. It is also interesting to notice that the evolutionary methods, regardless of the weighted mean strategy, only outperformed the SM approach on the  $S^{30}$  repository, since the low amount of mappings in  $S^5$  did not allow an effective weighted mean computation.

## 5 Conclusions and Further Work

In this paper, we presented a method for computing the GM correspondence based on an edit distance for correspondences called CED, which is a generalisation of a method based on the HD. Experimental validation shows that this approach is the best option to find the exact GM in three different correspondence scenarios, considering that by using the CED, a better represented GM is obtained at the cost of a larger computational complexity, especially as the number of mappings in correspondences increases. As future work, we are interested in comparing our method with more options for the GM calculation, putting particular emphasis in embedding approaches. It is also necessary to perform more experiments on real life repositories which contain structures and correspondences.

**Acknowledgment.** This research is supported by the Spanish projects TIN2016-77836-C2-1-R, ColRobTransp MINECO DPI2016-78957-R AEI/FEDER EU and the European project AEROARMS, H2020-ICT-2014-1-644271.

## References

1. Bille, P.: A survey on tree edit distance and related problems. *Theor. Comput. Sci.* **337**(1–3), 217–239 (2005)
2. Bunke, H., Günter, S.: Weighted mean of a pair of graphs. *Computing* **67**(3), 209–224 (2001)
3. Bunke, H., Jiang, X., Abegglen, K., Kandel, A.: On the weighted mean of a pair of strings. *Pattern Anal. Appl.* **5**(1), 23–30 (2002)
4. Caetano, T.S., McAuley, J.J., Cheng, L., Le, Q.V., Smola, A.J.: Learning graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(6), 1048–1058 (2009)
5. Cortés, X., Moreno, C., Serratos, F.: Improving the correspondence establishment based on interactive homography estimation. In: Wilson, R., Hancock, E., Bors, A., Smith, W. (eds.) CAIP 2013. LNCS, vol. 8048, pp. 457–465. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-40246-3\\_57](https://doi.org/10.1007/978-3-642-40246-3_57)

6. Cortés, X., Serratosa, F.: Learning graph matching substitution weights based on the ground truth node correspondence. *Int. J. Pattern Recogn. Artif. Intell.* **30**(02), 1650005 (2016)
7. Cortés, X., Serratosa, F., Moreno-García, C.F.: Semi-automatic pose estimation of a fleet of robots with embedded stereoscopic cameras. In: *Emerging Technologies and Factory Automation* (2016)
8. Ferrer, M., Valveny, E., Serratosa, F., Riesen, K., Bunke, H.: Generalized median graph computation by means of graph embedding in vector spaces. *Pattern Recogn.* **43**(4), 1642–1655 (2010)
9. Franek, L., Jiang, X.: Evolutionary weighted mean based framework for generalized median computation with application to strings. In: Gimelfarb, G., et al. (eds.) *SSPR & SPR*, pp. 70–78. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-34166-3\\_8](https://doi.org/10.1007/978-3-642-34166-3_8)
10. Franek, L., Jiang, X.: Ensemble clustering by means of clustering embedding in vector spaces. *Pattern Recogn.* **47**(2), 833–842 (2014)
11. Franek, L., Jiang, X., He, C.: Weighted mean of a pair of clusterings. *Pattern Anal. Appl.* **17**(1), 153–166 (2014)
12. Gao, X., Xiao, B., Tao, D., Li, X.: A survey of graph edit distance. *Pattern Anal. Appl.* **13**(1), 113–129 (2010)
13. Jiang, X., Bunke, H.: Learning by generalized median concept. In: Wang, P.S.-P. (ed), *Pattern Recognition and Machine Vision*, Chap. 15, pp. 231–246. River Publishers (2010)
14. Jiang, X., Wentker, J., Ferrer, M.: Generalized median string computation by means of string embedding in vector spaces. *Pattern Recogn. Lett.* **33**(7), 842–852 (2012)
15. Jonker, R., Volgenant, A.: A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing* **38**(4), 325–340 (1987)
16. Kuhn, H.W.: The Hungarian method for the assignment problem. *Naval Res. Log. Q.* **2**, 83–97 (1955)
17. Moreno-García, C.F., Cortés, X., Serratosa, F.: A graph repository for learning error-tolerant graph matching. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) *S+SSPR 2016*. LNCS, vol. 10029, pp. 519–529. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_46](https://doi.org/10.1007/978-3-319-49055-7_46)
18. Moreno-García, C.F., Serratosa, F.: Online learning the consensus of multiple correspondences between sets. *Knowl.-Based Syst.* **90**, 49–57 (2015)
19. Moreno-García, C.F., Serratosa, F.: Consensus of multiple correspondences between sets of elements. *Comput. Vis. Image Underst.* **142**, 50–64 (2016)
20. Moreno-García, C.F., Serratosa, F.: Obtaining the consensus of multiple correspondences between graphs through online learning. *Pattern Recogn. Lett.* **87**, 79–86 (2017)
21. Moreno-García, C.F., Serratosa, F.: Correspondence consensus of two sets of correspondences through optimisation functions. *Pattern Anal. Appl.* **20**(1), 201–213 (2017)
22. Moreno-García, C.F., Serratosa, F., Cortés, X.: Generalised median of a set of correspondences based on the hamming distance. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) *S+SSPR 2016*. LNCS, vol. 10029, pp. 507–518. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_45](https://doi.org/10.1007/978-3-319-49055-7_45)
23. Moreno-García, C.F., Serratosa, F., Jiang, X.: An edit distance between graph correspondences. In: Foggia, P., Liu, C.-L., Vento, M. (eds.) *GbrRPR 2017*. LNCS, vol. 10310, pp. 232–241. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-58961-9\\_21](https://doi.org/10.1007/978-3-319-58961-9_21)

24. Munkres, J.: Algorithms for the assignment and transportation problems. *J. Soc. Ind. Appl. Math.* **5**(1), 32–38 (1957)
25. Navarro, G.: A guided tour to approximate string matching. *ACM Comput. Surv.* **33**(1), 31–88 (2001)
26. Sanfeliu, A., Fu, K.: A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man Cybern. SMC* **13**(3), 353–362 (1983)
27. Serratoso, F.: Fast computation of bipartite graph matching. *Pattern Recogn. Lett.* **45**, 244–250 (2014)
28. Solé-Ribalta, A., Serratoso, F., Sanfeliu, A.: On the graph edit distance cost: properties and applications. *Int. J. Pattern Recogn. Artif. Intell.* **26**(05), 1260004 (2012)
29. Vento, M.: A long trip in the charming world of graphs for pattern recognition. *Pattern Recogn.* **48**(2), 291–301 (2015)
30. Wagner, R.A., Fischer, M.J.: The string-to-string correction problem. *J. ACM* **21**(1), 168–173 (1974)
31. Zhou, F., De La Torre, F.: Factorized graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.* **38**(9), 1774–1789 (2016)
32. Zitová, B., Flusser, J.: Image registration methods: a survey. *Image Vis. Comput.* **21**(11), 977–1000 (2003)



# Learning the Sub-optimal Graph Edit Distance Edit Costs Based on an Embedded Model

Pep Santacruz and Francesc Serratosà<sup>(✉)</sup>

Universitat Rovira i Virgili, Tarragona, Catalonia, Spain  
{joseluis.santacruz, francesc.serratosà}@urv.cat

**Abstract.** Graph edit distance has become an important tool in structural pattern recognition since it allows us to measure the dissimilarity of attributed graphs. One of its main constraints is that it requires an adequate definition of edit costs, which eventually determines which graphs are considered similar. These edit costs are usually defined as concrete functions or constants in a manual fashion and little effort has been done to learn them. The present paper proposes a framework to define these edit costs automatically. Moreover, we concretise this framework in two different models based on neural networks and probability density functions.

**Keywords:** Graph edit distance · Edit costs · Neural network  
Probability density function

## 1 Introduction

Graph edit distance [1, 2] is the most well-known and used distance between attributed graphs. It is defined as the minimum amount of required distortion to transform one graph into another. To this end, a number of distortion or edit functions consisting of deletion, insertion, and substitution of nodes and edges are defined. The basic idea is to assign an edit cost to each edit operation according to the amount of distortion that it introduces in the transformation to quantitatively evaluate the edit operations.

However, the structural and semantic dissimilarity of graphs is only correctly reflected by graph edit distance if the underlying edit costs are defined appropriately. For this reason, several methods have been presented to learn these costs. Most of them assume the substitution costs are weighted Euclidean distances and learn the weighting parameters [3–5]. Another one, [6], considers the insertion and deletion costs as constants and then applies optimisation techniques to tune these parameters. There are two other papers that define the edit costs as functions. The first one introduces a probabilistic model of the distribution of graph edit operations that allows them to derive edit costs [7]. The second paper is based on a self-organising map model [8] in which the edit costs are the output of a neural network. In both papers, the learning set is composed of classified graphs and the edit costs are optimised with regard to Dunn's index.

In the first part of this paper, we present a general model to learn the functions that define edit costs of the graph edit distance. This model opens the door to some techniques to learn these costs. In the second part of the paper, we present two concretisations of this

model. The first one is based on a probability density model learned through a multi-distribution Gaussian; the second one is based on a linear model learned through a neural net. The main difference between our model and the ones defined in [7, 8] is that in our model, the edit functions are learned using a local structure of the graphs but in the other ones, the edit functions are learned using only the attributes of the nodes or edges themselves.

This paper is structured as follows; in Sect. 2, we define the attributed graphs and the graph edit distance. In Sect. 3, we explain our learning model and in Sect. 4, we move to explain the embedding domain. Section 5 concretises two options of the presented learning model. Finally, Sect. 6 shows the experimental evaluation and Sect. 7 concludes the paper.

## 2 Attributed Graphs and Graph Edit Distance

Let  $G = (\Sigma_v, \Sigma_e, \gamma_v, \gamma_e)$  be an attributed graph representing an object.  $\Sigma_v = \{v_i | i = 1, \dots, n\}$  is the set of nodes and  $\Sigma_e = \{e_{xy} | x, y \in 1, \dots, n\}$  is the set of edges. With the aim of properly defining the graph matching, these sets are extended with null nodes and edges to be a complete graph of order  $n$ . We refer to null nodes of  $G$  by  $\hat{\Sigma}_v \subseteq \Sigma_v$  and we refer to null edges of  $G$  by  $\hat{\Sigma}_e \subseteq \Sigma_e$ . Functions  $\gamma_v : \Sigma_v \rightarrow \mathbb{R}^N$  and  $\gamma_e : \Sigma_e \rightarrow \mathbb{R}^M$  assign  $N$  attribute values to nodes and  $M$  attribute values to edges.

We also define the *star* of a node  $v_a$ , named  $S_a$ , on an attributed graph  $G$ , as another graph  $S_a = (\Sigma_v^{S_a}, \Sigma_e^{S_a}, \gamma_v^{S_a}, \gamma_e^{S_a})$ .  $S_a$  has the structure of an attributed graph but it is only composed of nodes connected to  $v_a$  by an edge and these connecting edges. Formally,  $\Sigma_v^{S_a} = \{v_b | e_{ab} \in \Sigma_e - \hat{\Sigma}_e\}$  and  $\Sigma_e^{S_a} = \{e_{ab} \in \Sigma_e - \hat{\Sigma}_e\}$ . Finally,  $\gamma_v^{S_a}(v_b) = \gamma_v(v_b)$ ,  $\forall v_b \in \Sigma_v^{S_a}$  and  $\gamma_e^{S_a}(e_{ab}) = \gamma_e(e_{ab})$ ,  $\forall e_{ab} \in \Sigma_e^{S_a}$ .

Given two attributed graphs  $G$  and  $G'$ , and a correspondence  $f$  between them, the graph edit cost, represented by the expression  $EditCost(G, G', f)$ , is the cost of the edit operations that the correspondence  $f$  imposes. It is based on adding the functions:

- $C_{vs}$  is a distance that represents the cost of substituting node  $v_a$  of  $G$  by node  $f(v_a)$  of  $G'$ .
- $C_{es}$  is a distance that represents the cost of substituting edge  $e_{ab}$  of  $G$  by edge  $e'_{ij}$  of  $G'$ .  $f(v_a) = v'_i$  and  $f(v_b) = v'_j$ .
- $C_{vd}$  is the cost of deleting node  $v_a$  of  $G$  (mapping it to a null node).
- $C_{vi}$  is the cost of inserting node  $v'_i$  of  $G'$  (being mapped from a null node).
- $C_{ed}$  is the cost of assigning edge  $e_{ab}$  of  $G$  to a null edge of  $G'$ .
- $C_{ei}$  is the cost of assigning edge  $e'_{ij}$  of  $G'$  to a null edge of  $G$ .

For the cases in which two null nodes or two null edges are mapped, this cost is 0. Then, the graph edit distance, GED, is defined as the minimum cost under any possible bijective correspondence  $f$  in the set  $F$ , which is composed of all bijective correspondences between  $G$  and  $G'$

$$GED(G, G') = \min_{f \in F} \{EditCost(G, G', f)\}. \tag{1}$$

If we consider  $f(v_a) = v'_i$  and  $f(v_b) = v'_j$ , the *EditCost* is,

$$\begin{aligned}
 EditCost(G, G', f) = & \\
 & \sum_{\forall v_a \in \Sigma_v - \hat{\Sigma}_v, s.t. v'_i \in \Sigma'_v - \hat{\Sigma}'_v} C_{vs}(v_a, v'_i) + \sum_{\forall e_{ab} \in \Sigma_e - \hat{\Sigma}_e, s.t. e'_{ij} \in \hat{\Sigma}'_e - \hat{\Sigma}'_e} C_{es}(e_{ab}, e'_{ij}) + \\
 & \sum_{\forall v_a \in \Sigma_v - \hat{\Sigma}_v, s.t. v'_i \in \hat{\Sigma}'_v} C_{vd}(v_a) + \sum_{\forall e_{ab} \in \Sigma_e - \hat{\Sigma}_e, s.t. e'_{ij} \in \hat{\Sigma}'_e} C_{ed}(e_{ab}) + \\
 & \sum_{\forall v_a \in \hat{\Sigma}_v, s.t. v'_i \in \Sigma'_v - \hat{\Sigma}'_v} C_{vi}(v'_i) + \sum_{\forall e_{ab} \in \hat{\Sigma}_e, s.t. e'_{ij} \in \Sigma'_e - \hat{\Sigma}'_e} C_{ei}(e'_{ij})
 \end{aligned} \tag{2}$$

We define the optimal correspondence  $\hat{f}$  as the one that obtains the minimum *EditCost*( $G, G', \hat{f}$ ).

### 2.1 Sub-optimal Computation of the Graph Edit Distance

The optimal computation of the GED is usually carried out by means of the A\* algorithm [11, 12]. Unfortunately, the computational complexity of these methods is exponential in the number of nodes of the involved graphs. For this reason, several sub-optimal methods to compute the GED have been presented. The main idea is to optimise local criteria instead of global criteria [9, 10] and therefore a sub-optimal GED can be computed in polynomial time. To this end, the Edit Cost between two graphs (Eq. 1) is the addition of the costs of mapping their local structures:

$$\begin{aligned}
 EditCost^{sub}(G, G', f) = & \sum_{\forall v_a \in \Sigma_v - \hat{\Sigma}_v, s.t. v'_i \in \Sigma'_v - \hat{\Sigma}'_v} C^s(S_a, S'_i) \\
 & + \sum_{\forall v_a \in \Sigma_v - \hat{\Sigma}_v, s.t. v'_i \in \hat{\Sigma}'_v} C^d(S_a) + \sum_{\forall v_a \in \hat{\Sigma}_v, s.t. v'_i \in \Sigma'_v - \hat{\Sigma}'_v} C^i(S'_i)
 \end{aligned} \tag{3}$$

Where  $f(v_a) = v'_i$ . Besides,  $C^s$  denotes the cost of substituting the star  $S_a$  centred at node  $v_a$  by the star  $S'_i$  centred at node  $v'_i$ .  $C^d$  denotes the cost of deleting the star  $S_a$  and  $C^i$  denotes the cost of inserting the star  $v'_i$ . These costs depend on the structure of the stars and also on the costs on nodes and edges:  $C_{vs}$ ,  $C_{vd}$ ,  $C_{vi}$ ,  $C_{es}$ ,  $C_{ed}$  and  $C_{ei}$ . These costs are computed in the same way as it is done with graphs, since stars are defined as graphs with a concrete structure.

Similarly to the optimal GED, we define the sub-optimal edit distance as the minimum of the edit cost:

$$GED^{sub}(G, G') = \min_{f \in F} \{EditCost^{sub}(G, G', f)\} \tag{4}$$

And also, we define  $\hat{f}^{sub}$  as the correspondence in  $F$  such that  $EditCost^{sub}(G, G', \hat{f}^{sub})$  is the minimum one.



If the ground truth correspondence  $\hat{f}^p$  imposes two nodes to be substituted then it may hold that the substitution cost of the involved stars might be lower than the substitution costs of the combinations of the other stars. Moreover, if the ground truth correspondence  $\hat{f}^p$  imposes a node to be deleted then it may hold that the deletion cost of the involved star might be lower than the deletion costs of the stars that the ground truth correspondence imposes they have to be substituted. Similarly occurs with the node insertions. This method was used in [13].

Figure 2 shows an example of a ground truth correspondence  $\hat{f}^p$ . It may happen that  $C^s(S_1^p, S_1^{p'})$  would have to be lower than  $C^s(S_1^p, S_2^{p'})$  and  $C^s(S_2^p, S_1^{p'})$ . Similarly occurs with  $C^s(S_2^p, S_2^{p'})$ . Moreover, it may happen that  $C^d(S_3^p)$  would have to be lower than  $C^d(S_1^p)$  and  $C^d(S_2^p)$ . Similarly occurs with  $C^d(S_4^p)$ . Finally, it also may happen that  $C_i(S_3^{p'})$  would have to be lower than  $C_i(S_1^{p'})$  and  $C_i(S_2^{p'})$ . The same for  $C_i(S_2^{p'})$ .

To fix these initial ideas into a learning model, we have defined two classes of mappings in the substitution cases; two other classes of mappings in the deletion cases; and another two classes of mappings in the insertion cases.

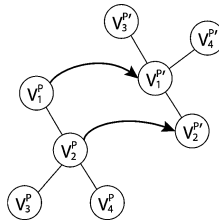
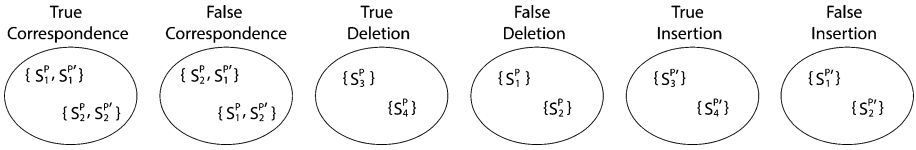


Fig. 2. Ground-truth correspondence  $\hat{f}^p$  from  $G^p$  to  $G^{p'}$ .

If a ground-truth correspondence  $\hat{f}^p$  defines the mapping  $\hat{f}^p(v_a^p) = v_i^{p'}$  between non-null nodes then we say that the pair of stars  $\{S_a^p, S_i^{p'}\}$  belongs to class *True\_Substitution*. Contrarily, all combinations of pairs  $\{S_a^p, S_j^{p'}\}$  that  $j \neq i$  and also all combination of pairs  $\{S_b^p, S_i^{p'}\}$  that  $b \neq a$  between non-null nodes belong to class *False\_Substitution*. Moreover, if the ground-truth correspondence  $\hat{f}^p$  imposes the node  $v_a^p$  has to be deleted, then we consider that the star  $S_a^p$  belongs to class *True\_Deletion*. Contrarily, all stars  $S_b^p$  such that their central nodes  $v_b^p$  are substituted, (nodes  $v_b^p$  such that  $\hat{f}^p(v_b^p) = v_j^{p'}, b \neq a$ ), belong to class *False\_Deletion*. Similarly occurs with the insertion operations. If the ground-truth correspondence  $\hat{f}^p$  imposes the node  $v_i^{p'}$  has to be inserted, then we consider that the star  $S_i^{p'}$  belongs to class *True\_Insertion*. Contrarily, all stars  $S_j^{p'}$  such that their central nodes  $v_j^{p'}$  are substituted (all nodes such that  $\hat{f}^p(v_b^p) = v_j^{p'}, j \neq i$ ) belong to class *False\_Insertion*.



Figure 3 shows the classes of pairs of stars previously defined, given the substitutions, deletions and insertions of the example in Fig. 2.



**Fig. 3.** Classes and mappings given example in Fig. 2.

We proceed to formalise the definition of these six sets. Suppose that we have  $L$  pairs of graphs  $(G^p, G^{p'})$ ,  $1 \leq p \leq L$ , together with their ground-truth correspondences  $\hat{f}^p$ . Then for all correspondences  $\hat{f}^p$  and for all node-to-node mappings  $\hat{f}^p(v_a^p) = v_i^{p'}$  we set,

$$\begin{aligned}
 \{S_a^p, S_i^{p'}\} &\in \text{True\_Substitution} \text{ if } v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p \text{ and } v_i^{p'} \in \Sigma_v^{p'} - \hat{\Sigma}_v^{p'} \\
 \{S_a^p, S_k^{p'}\} &\in \text{False\_Substitution} \text{ if } k \neq i \text{ and } v_j^{p'} \in \Sigma_v^{p'} - \hat{\Sigma}_v^{p'} \\
 \{S_b^p, S_i^{p'}\} &\in \text{False\_Substitution} \text{ if } b \neq a \text{ and } v_b^p \in \Sigma_v^p - \hat{\Sigma}_v^p \\
 \{S_a^p\} &\in \text{True\_Deletion} \text{ if } v_a^p \in \hat{\Sigma}_v^p \\
 \{S_a^p\} &\in \text{False\_Deletion} \text{ if } v_a^p \in \Sigma_v^p - \hat{\Sigma}_v^p \\
 \{S_i^{p'}\} &\in \text{True\_Insertion} \text{ if } v_i^{p'} \in \hat{\Sigma}_v^{p'} \\
 \{S_i^{p'}\} &\in \text{False\_Insertion} \text{ if } v_i^{p'} \in \Sigma_v^{p'} - \hat{\Sigma}_v^{p'}
 \end{aligned} \tag{5}$$

## 4 Embedding Stars into Vectors

The aim of this paper is to present a model to learn costs  $C^s$ ,  $C^d$  and  $C^i$  based on a classical machine-learning method. To do so, we need these costs to be modelled as functions, in which the domain is a point in a vector space and the codomain is a Real number. Therefore, we have to map the stars to points in a suitable vector space. This mapping has to encode the stars by equal size vectors and produce one vector per star. Mathematically, for a given star  $S_a$ , our star embedding is a function  $\Phi$ , which maps  $S_a$  to a point  $E_a$  in a  $T$  dimension space  $\mathbb{R}^T$ . It is given as  $\Phi(S_a) = E_a$ . The value  $T$  is concretised above.

Figure 4 graphically shows the embedding of the star  $S_a$ . The first  $N$  elements are the attributes on the nodes and the next one is the number of nodes of the star,  $n_{S_a}$ . The next cells are filled by the histograms generated by the attributes of the external nodes and the attributes of the external edges. Histograms  $h_{r(i)}$  and  $h_{e(i)}$  represent histograms generated by the  $i^{\text{th}}$  attribute of the nodes and edges, respectively.  $N$  and  $M$  are the number of attributes on the nodes and edges, respectively. Finally,  $\tilde{N}$  and  $\tilde{M}$  are the number of bins of the node and edge histograms, respectively. This representation has been inspired by the one presented in [14]. In that case, the model embedded a whole

graph into a vector. Since we want to embed a star, which is a special structure of a graph, we have somewhat concretised the embedding model. Thus,  $T = N + 1 + \tilde{N} * N + \tilde{M} * M$ .

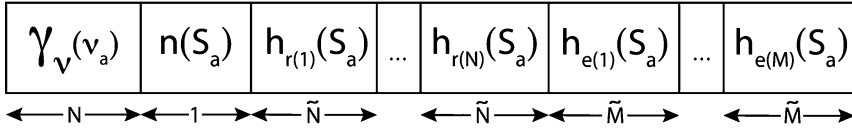


Fig. 4. The  $E_a$  embedding of star  $S_a$ .

Then, given the six sets, our method defines three matrices as shown in Fig. 5. The *Substitution\_Matrix* has three sets of columns. The first two ones have the embedded stars  $E_a$  and  $E'_i$  that their pairs of stars are in the sets *True\_Substitution* or *False\_Substitution*. The third set is composed of only one column that has ones and zeros. A zero in this column informs the pair of stars belongs to the *True\_Substitution* set and a one informs that it belongs to the *False\_Substitution* set. The *Deletion\_Matrix* has two sets of columns:  $E_a$  and a column of ones and zeros. A zero in this column informs the star  $S_a$  belongs to the *True\_Deletion* set and a one informs that it belongs to the *False\_Deletion* set. Similarly occurs with the *Insertion\_Matrix* but considering the stars  $S'_i$  of the other graph.

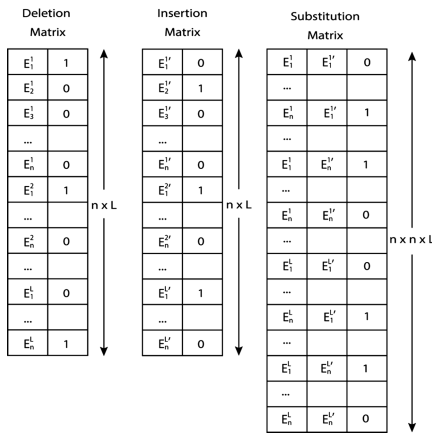


Fig. 5. The  $E_a$  embedding of star  $S_a$ .

Then, we define the substitution, deletion and insertion functions as the output of a machine learning method using these matrices as follows:

$$C^s = \text{Machine\_Learning}(\textit{Substitution\_Matrix})$$

$$C^d = \text{Machine\_Learning}(\textit{Deletion\_Matrix})$$

$$C^i = \text{Machine\_Learning}(\textit{Insertion\_Matrix}).$$

## 5 Graph Matching Algorithm and Learning Methods

In the previous sections, we have presented a general framework to learn the edit functions. Although this framework could be concretised into different methods, we present, in this section, only two different examples. Moreover, several graph-matching algorithms could be adapted to use these edit functions. In the experimental evaluation, we computed the graph distance through the bipartite graph-matching algorithm [9]. In this case, adapting the algorithm only means how  $C^s$ ,  $C^d$  and  $C^i$  are defined in the first step of the algorithm (Sect. 2). In the original definition of the algorithm [9], these costs were computed considering that stars are graphs with a concrete structure. In the next two sub-sections, we show how we deduce these costs.

### 5.1 Neural Network

We model  $C^s$  by a regression function learned through an artificial neural network,  $nn^s$ , given the *Substitution\_Matrix*. When the neural net has learned the regression function, the substitution cost  $C^s(S_a, S'_i)$  is computed as the output of this neural network,  $nn^s$ , as follows:

$$C^s(S_a, S'_i) = \textit{Output}(nn^s, [E_a, E'_i]) \quad (6)$$

We also model  $C^d$  by a regression function based on an artificial neural network,  $nn^d$ , learned from *Deletion\_Matrix*, in a similar way than  $C^s$ . Nevertheless, in this case, we only use the information of the first graph. Then, we have,

$$C^d(S_a) = \textit{Output}(nn^d, E_a) \quad (7)$$

Similarly occurs with the insertion cost but using the information of the second graph. We model  $C^i$  by an artificial neural network,  $nn^i$ , learned from *Insertion\_Matrix*. Then, we have,

$$C^i(S'_i) = \textit{Output}(nn^i, E'_i) \quad (8)$$

### 5.2 Probability Density Distribution

We define  $C^s$  by two probability density functions based on a mixture of Gaussians,  $pdf\_true^s$  and  $pdf\_false^s$ . The first density function is modelled by columns that have the information about  $E_a$  and  $E'_i$  in the *Substitution\_Matrix*, but with only the rows that

have a 1 in the last column. The second density function is modelled in a similar way but with only the rows that have a 0 in the last column.

Thus, the substitution cost  $C^s(S_a, S'_i)$  is defined as the subtraction of the probabilities obtained from these probability density functions (Eq. 9). Constant 1 is needed to assure the cost is always positive. We want the cost to be low if the probability obtained from the set *True\_Substitution* is high or the probability obtained from the set *False\_Substitution* is low.

$$C^s(S_a, S'_i) = 1 - Prob(pdf\_true^s, [E_a, E'_i]) + Prob(pdf\_false^s, [E_a, E'_i]) \quad (9)$$

Functions  $C^d$  and  $C^i$  are modelled in a similar way. Nevertheless, matrices *Deletion\_Matrix* and *Insertion\_Matrix* are used. Thus, we have:

$$C^d(S_a) = 1 - Prob(pdf\_true^d, E_a) + Prob(pdf\_false^d, E_a) \quad (10)$$

$$C^i(S'_i) = 1 - Prob(pdf\_true^i, E'_i) + Prob(pdf\_false^i, E'_i) \quad (11)$$

## 6 Experimental Evaluation

The presented method has been validated using four databases in the public graph repository *Tarragona\_Graphs* presented in [15]. The main characteristic of this repository is that its registers are not only composed of a graph and its class, but composed of a pair of graphs and a ground-truth matching between them, as well as their class. This register structure is useful to analyse and develop graph-matching algorithms and to learn their parameters in a broad manner.

Table 1 shows the accuracy (in bold the highest scores) computed by the Bipartite graph matching and the Learning Bipartite graph matching (our proposal). In the first case, we have considered the Degree and the Star as a local structure. In the second case, we have considered the Neural Network (Sect. 5.1) and the Probability density function (Sect. 5.2). In the case of the Neural Network, we have tested the embedding presented in Fig. 4 and also a reduced embedding in which the histogram of the neighbours' attributes has not been considered. Note that depending on the number of nodes and the number of bins per attribute, this information of the embedding is the part that could take more space. The Neural Networks have been configured with only one hidden layer that have half of the width of the input layer. The probability density functions have been configured as multimodal Gaussians. In the case of *Letter High* and *Letter Med*, we used two modal and in the case of the *Letter Low*, only one modal. The *House Hotel* database always returned "ill condition".

Star configuration returns higher accuracies than Degree configuration, as reported in other papers. The neural network returns the highest accuracies and it seems as the histogram information positively contributes to the embedding model since there is an important reduction on the accuracy if it is discarded.

**Table 1.** Accuracy of four databases in *Tarragona Graphs* repository given the original Bipartite graph matching and the Learning Bipartite graph matching (our proposal). We have considered several configurations.

Algorithm	Configuration	Letter high	Letter med	Letter low	House hotel
Original	Star	0.89	<b>0.90</b>	0.97	0.88
Bipartite	Degree	0.87	0.85	0.97	0.71
Learning	NN	<b>0.91</b>	<b>0.90</b>	<b>0.98</b>	0.98
Bipartite	NN (No histogram)	0.89	0.87	0.97	<b>0.99</b>
	Prob. density function	0.83	0.76	0.93	Ill condition

## 7 Conclusions

Edit costs functions are application dependent and usually set manually based on maximising the accuracy in the recognition process. We have proposed a general framework to learn the substitution, deletion and insertion costs based on reducing the hamming distance between the deduced correspondences and the ground-truth correspondences. Moreover, we have concretised our framework on two models, one based on neural networks and the other one based on multimodal probability density functions.

We have tested our framework on four public databases and we have empirically deduced that the neural network achieves the highest accuracies, therefore, it seems to be worth learning these costs.

**Acknowledgments.** This research is supported by the Spanish projects TIN2016-77836-C2-1-R and ColRobTransp MINECO DPI2016-78957-R AEI/FEDER EU; and also, the European project AEROARMS, H2020-ICT-2014-1-644271.





## References

1. Bunke, H., Allermann, G.: Inexact graph matching for structural pattern recognition. *Pattern Recogn. Lett.* **1**(4), 245–253 (1983)
2. Sanfeliu, A., Fu, K.S.: A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man Cybern.* **13**(3), 353–362 (1983)
3. Caetano, T., et al.: Learning graph matching. *Trans. Pattern Anal. Mach. Intell.* **31**(6), 1048–1058 (2009)
4. Leordeanu, M., Sukthankar, R., Hebert, M.: Unsupervised learning for graph matching. *Int. J. Comput. Vis.* **96**(1), 28–45 (2012)
5. Cortés, X., Serratos, F.: Learning graph matching substitution weights based on the ground truth node correspondence. *Int. J. Pattern Recogn. Artif. Intell.* **30**(2), 1650005 (2016). [22 pages]
6. Cortés, X., Serratos, F.: Learning graph-matching edit-costs based on the optimality of the Oracle’s node correspondences. *Pattern Recogn. Lett.* **56**, 22–29 (2015)
7. Neuhaus, M., Bunke, H.: Automatic learning of cost functions for graph edit distance. *Inf. Sci.* **177**(1), 239–247 (2007)

8. Neuhaus, M., Bunke, H.: Self-organizing maps for learning the edit costs in graph matching. *IEEE Trans. Syst. Man Cybern. Part B* **35**(3), 503–514 (2005)
9. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**(7), 950–959 (2009)
10. Serratosa, F.: Fast computation of bipartite graph matching. *Pattern Recogn. Lett.* **45**, 244–250 (2014)
11. Hart, P., Nilsson, N., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **4**(2), 100–107 (1968)
12. Ferrer, M., Serratosa, F., Riesen, K.: Improving bipartite graph matching by assessing the assignment confidence. *Pattern Recogn. Lett.* **65**, 29–36 (2015)
13. Serratosa, F., Cortés, X.: Interactive graph-matching using active query strategies. *Pattern Recogn.* **48**(4), 1364–1373 (2015)
14. Luqman, M.M., Ramel, J.-Y., Lladós, J., Brouard, T.: Fuzzy multilevel graph embedding. *Pattern Recogn.* **46**(2), 551–565 (2013)
15. Moreno-García, C.F., Cortés, X., Serratosa, F.: A graph repository for learning error-tolerant graph matching. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) *S+SSPR 2016*. LNCS, vol. 10029, pp. 519–529. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_46](https://doi.org/10.1007/978-3-319-49055-7_46)



# Ring Based Approximation of Graph Edit Distance

David B. Blumenthal<sup>1</sup><sup>(✉)</sup>, Sébastien Bougleux<sup>2</sup>, Johann Gamper<sup>1</sup>,  
and Luc Brun<sup>2</sup>

<sup>1</sup> Faculty of Computer Science, Free University of Bozen-Bolzano, Bolzano, Italy

{david.blumenthal,gamper}@inf.unibz.it

<sup>2</sup> Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, Caen, France  
bougleux@unicaen.fr, luc.brun@ensicaen.fr

**Abstract.** The graph edit distance (GED) is a flexible graph dissimilarity measure widely used within the structural pattern recognition field. A widely used paradigm for approximating GED is to define local structures rooted at the nodes of the input graphs and use these structures to transform the problem of computing GED into a linear sum assignment problem with error correction (LSAPE). In the literature, different local structures such as incident edges, walks of fixed length, and induced subgraphs of fixed radius have been proposed. In this paper, we propose to use rings as local structure, which are defined as collections of nodes and edges at fixed distances from the root node. We empirically show that this allows us to quickly compute a tight approximation of GED.

**Keywords:** Graph edit distance · Graph matching · Upper bounds

## 1 Introduction

Due to the flexibility and expressiveness of labeled graphs, graph representations of objects such as molecules and shapes are widely used for addressing pattern recognition problems. For this, a graph (dis-)similarity measure has to be defined. A widely used measure is the graph edit distance (GED), which equals the minimum cost of a sequence of edit operations transforming one graph into another. As exactly computing GED is *NP*-hard [17], research has mainly focused on the design of approximative heuristics that quickly compute upper bounds for GED. The development of such heuristics was particularly triggered by the introduction of the paradigm *LSAPE-GED*, which transforms GED to the linear sum assignment problem with error correction (LSAPE) [10, 17]. *LSAPE* extends the linear sum assignment problem by allowing rows and columns to be not only substituted, but also deleted and inserted. *LSAPE-GED* works as follows: In a first step, the graphs  $G$  and  $H$  are decomposed into local structures rooted at their nodes. Next, a distance measure between these local structures is defined. This measure is used to populate an instance of *LSAPE*, whose rows and columns correspond to the nodes of  $G$  and  $H$ , respectively. Finally, the constructed *LSAPE*

instance is solved. The computed solution is interpreted as a sequence of edit operations, whose cost is returned as an upper bound for  $\text{GED}(G, H)$ .

The original instantiations BP [10] and STAR [17] of LSAPÉ-GED define the local structure of a node as, respectively, the set of its incident edges and the set of its incident edges together with the terminal nodes. Since then, further instantiations have been proposed. Like BP, the algorithms BRANCH-UNI [18], BRANCH, and BRANCH-FAST [2] use the incident edges as local structures. They differ from BP in that they use distance measures for the local structures that also allow to derive lower bounds for GED. In contrast to that, the algorithms SUBGRAPH [6] and WALKS [8] define larger local structures. Given a constant  $L$ , SUBGRAPH defines the local structure of a node  $u$  as the subgraph which is induced by the set of nodes that are within distance  $L$  from  $u$ , while WALKS defines it as the set of walks of length  $L$  starting at  $u$ . SUBGRAPH uses GED as the distance measure between its local structures and hence runs in polynomial time only if the input graphs have constantly bounded maximum degrees. Not all instantiations of LSAPÉ-GED are designed for general edit costs: STAR and BRANCH-UNI expect the edit costs to be uniform, and WALKS assumes that the costs of all edit operation types are constant. As an extension of LSAPÉ-GED, it has been suggested to define node centrality measures, transform the LSAPÉ instance constructed by any instantiation of LSAPÉ-GED such that assigning central to non-central nodes is penalized, and return the minimum of the edit costs induced by solutions to the original and the transformed instances as an upper bound for GED [12, 16].

Not all heuristics for GED follow the paradigm LSAPÉ-GED. Most notably, some methods use variants of local search to improve a previously computed upper bound [4, 7, 11, 14]. These methods yield tighter upper bounds than LSAPÉ-GED instantiations at the price of a significantly increased runtime, and use LSAPÉ-GED instantiations for initialization. They are thus no competitors of LSAPÉ-GED instantiations and will hence not be considered any further in this paper.

In this paper, we propose a new instantiation RING of LSAPÉ-GED that is similar to SUBGRAPH and WALKS in that it also uses local structures whose sizes are bounded by a constant  $L$ —namely, rings. Intuitively, the ring rooted at a node  $u$  is a collection of disjoint sets of nodes and edges which are within distances  $l < L$  from  $u$ . Experiments show that RING yields the tightest upper bound of all instantiations of LSAPÉ-GED. The advantage of rings w. r. t. subgraphs is that ring distances can be computed in polynomially. The advantage w. r. t. walks is that rings can model general edit costs, avoid redundancies due to multiple node or edges inclusions, and allow to define a fine-grained distance measure between the local structures. The rest of the paper is organized as follows: In Sect. 2, important concepts are introduced. In Sect. 3, RING is presented. In Sect. 4, the experimental results are summarized. Section 5 concludes the paper.

## 2 Preliminaries

In this paper, we consider undirected labeled graphs  $G = (V^G, E^G, \ell_V^G, \ell_E^G)$ , where  $V^G$  and  $E^G$  are sets of nodes and edges, and  $\ell_V^G : V^G \rightarrow \Sigma_V$ ,  $\ell_E^G : E^G \rightarrow \Sigma_E$



**Table 1.** Edit operations and edit costs for transforming a graph  $G$  into a graph  $H$ .

Edit operation	Edit cost	Short notation
Substitute node $u \in V^G$ by node $v \in V^H$	$c_V(\ell_V^G(u), \ell_V^H(u))$	$c_V(u, v)$
Delete isolated node $u \in V^G$ from $V^G$	$c_V(\ell_V^G(u), \epsilon)$	$c_V(u, \epsilon)$
Insert isolated node $v$ into $V^H$	$c_V(\epsilon, \ell_V^H(u))$	$c_V(\epsilon, v)$
Substitute edge $e \in E^G$ by edge $f \in E^H$	$c_E(\ell_E^G(e), \ell_E^H(f))$	$c_E(e, f)$
Delete edge $e \in E^G$ from $E_G$	$c_E(\ell_E^G(e), \epsilon)$	$c_E(e, \epsilon)$
Insert edge $f$ into $E^H$	$c_E(\epsilon, \ell_E^H(f))$	$c_E(\epsilon, f)$

are labeling functions. Furthermore, we are given non-negative edit cost functions  $c_V : \Sigma_V \cup \{\epsilon\} \times \Sigma_V \cup \{\epsilon\} \rightarrow \mathbb{R}_{\geq 0}$  and  $c_E : \Sigma_E \cup \{\epsilon\} \times \Sigma_E \cup \{\epsilon\} \rightarrow \mathbb{R}_{\geq 0}$ , where  $\epsilon$  is a special label reserved for dummy nodes and edges, and the equations  $c_V(\alpha, \alpha) = 0$  and  $c_E(\beta, \beta) = 0$  hold for all  $\alpha \in \Sigma_V \cup \{\epsilon\}$  and all  $\beta \in \Sigma_E \cup \{\epsilon\}$ . An edit path  $P$  between graphs  $G$  and  $H$  is a sequence of edit operations with non-negative edit costs defined in terms of  $c_V$  and  $c_E$  (Table 1) that transform  $G$  into  $H$ . Its cost  $c(P)$  is defined as the sum over the costs of its edit operations.

**Definition 1 (GED).** *The graph edit distance between graphs  $G$  and  $H$  is defined as  $\text{GED}(G, H) = \min_{P \in \Psi(G, H)} c(P)$ , where  $\Psi(G, H)$  is the set of all edit paths between  $G$  and  $H$ .*

The key insight behind the paradigm LSAPe-GED is that a complete set of node edit operations—i.e., a set of node edit operations that specifies for each node of the input graphs whether it has to be substituted, inserted, or deleted—can be extended to an edit path, whose edit cost is an upper bound for GED [3, 4, 17]. For constructing a set of node operations that induces a cheap edit path, a suitably defined instance of LSAPe is solved. LSAPe is defined as follows [5]:

**Definition 2 (LSAPe).** *Given a matrix  $\mathbf{C} = (c_{i,k}) \in \mathbb{R}_{\geq 0}^{(n+1) \times (m+1)}$  with  $c_{n+1, m+1} = 0$ , LSAPe consists in the task to compute an assignment  $\pi^* \in \arg \min_{\pi \in \Pi_{n,m}} \mathbf{C}(\pi)$ .  $\Pi_{n,m}$  is the set of assignments of rows of  $\mathbf{C}$  to columns of  $\mathbf{C}$  such that each row except for  $n+1$  and each column except for  $m+1$  is covered exactly once, and  $\mathbf{C}(\pi) = \sum_{i=1}^{n+1} \sum_{k \in \pi[i]} c_{i,k}$ .*

Instantiations of LSAPe-GED construct a LSAPe instance  $\mathbf{C}$  of size  $(|V^G| + 1) \times (|V^H| + 1)$ , such that the rows and columns of  $\mathbf{C}$  correspond to the nodes of  $G$  and  $H$  plus one dummy node used for representing insertions and deletions. A feasible solution for  $\mathbf{C}$  can hence be interpreted as a complete set of node edit operations, which induces an upper bound for GED. An optimal solution for  $\mathbf{C}$  can be found in  $O(\min\{n, m\}^2 \max\{n, m\})$  time [5]; greedy suboptimal solvers run in  $O(nm)$  time [13]. For populating  $\mathbf{C}$ , instantiations of LSAPe-GED associate the nodes  $u_i \in V^G$  and  $v_k \in V^H$  with local structures  $\mathcal{S}^G(u_i)$  and  $\mathcal{S}^H(v_k)$ , and then construct  $\mathbf{C}$  by setting  $c_{i,k} = d_{\mathcal{S}}(\mathcal{S}^G(u_i), \mathcal{S}^H(v_k))$ ,

$c_{i,|V^H|+1} = d_S(\mathcal{S}^G(u_i), \mathcal{S}(\epsilon))$ , and  $c_{|V^G|+1,k} = d_S(\mathcal{S}(\epsilon), \mathcal{S}^H(v_k))$ , where  $d_S$  is a distance measure for the local structures and  $\mathcal{S}(\epsilon)$  is a special local structure assigned to dummy nodes.

### 3 Ring Based Upper Bounds for GED

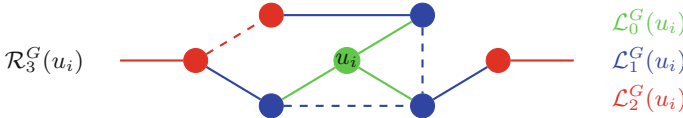
#### 3.1 Definition of Ring Structures and Ring Distances

Let  $u_i, u_j \in V^G$  be two nodes in  $G$ . The distance  $d_V^G(u_i, u_j)$  between the nodes  $u_i$  and  $u_j$  is defined as the number of edges of a shortest path connecting them or as  $\infty$  if they are in different connected components of  $G$ . The eccentricity of a node  $u_i \in V^G$  and the diameter of a graph  $G$  are defined as  $e_V^G(u_i) = \max_{u_j \in V^G} d_V^G(u_i, u_j)$  and  $\text{diam}(G) = \max_{u \in V^G} e_V^G(u)$ , respectively.

**Definition 3 (Ring, Layer, Outer Edges, Inner Edges).** *Given a constant  $L \in \mathbb{N}_{>0}$  and a node  $u_i \in V^G$ , we define the ring rooted at  $u_i$  in  $G$  as the sequence of disjoint layers  $\mathcal{R}_L^G(u_i) = (\mathcal{L}_l^G(u_i))_{l=0}^{L-1}$  (Fig. 1). The  $l^{\text{th}}$  layer rooted at  $u_i$  is defined as  $\mathcal{L}_l^G(u_i) = (V_l^G(u_i), OE_l^G(u_i), IE_l^G(u_i))$  where:*

- $V_l^G(u_i) = \{u_j \in V^G \mid d_V^G(u_i, u_j) = l\}$  is the set of nodes at distance  $l$  of  $u_i$ ,
- $IE_l^G(u_i) = E^G \cap (V_l^G(u_i) \times V_l^G(u_i))$  is the set of inner edges connecting two nodes in the  $l^{\text{th}}$  layer, and
- $OE_l^G(u_i) = E^G \cap (V_l^G(u_i) \times V_{l+1}^G(u_i))$  is the set of outer edges connecting a node in the  $l^{\text{th}}$  layer to a node in the  $(l + 1)^{\text{th}}$  layer.

For the dummy node  $\epsilon$ , we define  $\mathcal{R}_L(\epsilon) = ((\emptyset, \emptyset, \emptyset)_l)_{l=0}^{L-1}$ .



**Fig. 1.** Visualization of Definition 3. Inner edges are dashed, outer edges are solid.

*Remark 1 (Properties of Rings and Layers).* The first layer  $\mathcal{L}_0^G(u_i)$  of a node  $u_i$  corresponds to  $u_i$ 's local structure as defined by BP, BRANCH, BRANCH-FAST, and BRANCH-UNI. We have  $OE_l^G(u_i) = \emptyset$  just in case  $l > e_V^G(u_i) - 1$  and  $\mathcal{L}_l^G(u_i) = (\emptyset, \emptyset, \emptyset)$  just in case  $l > e_V^G(u_i)$ . Moreover, the identities  $E^G = \bigcup_{l=0}^{L-1} (OE_l^G(u_i) \cup IE_l^G(u_i))$  and  $V^G = \bigcup_{l=0}^{L-1} V_l^G(u_i)$  hold for all  $u_i \in V^G$  just in case  $L > \text{diam}(G)$ .

In our instantiation RING of LSAPG-GED, we use rings as local structures, i.e., define  $\mathcal{S}^G(u_i) = \mathcal{R}_L^G(u_i)$ . The next step is to define a distance measure  $d_{\mathcal{R}}$  that maps two rings to a non-negative real number. For doing so, we first define a measure  $d_{\mathcal{L}}$  that returns the distance between two layers. So let  $\mathcal{L}_l^G(u)$

and  $\mathcal{L}_l^H(v)$  be the  $l^{\text{th}}$  layers rooted at nodes  $u \in V^G \cup \{\epsilon\}$  and  $v \in V^H \cup \{\epsilon\}$ , respectively. Then  $d_{\mathcal{L}}$  is defined as

$$d_{\mathcal{L}}(\mathcal{L}_l^G(u), \mathcal{L}_l^H(v)) = \alpha_0 \phi_V(V_l^G(u), V_l^H(v)) + \alpha_1 \phi_E(OE_l^G(u), OE_l^H(v)) + \alpha_2 \phi_E(IE_l^G(u), IE_l^H(v)),$$

where  $\phi_V : \mathcal{P}(V^G) \times \mathcal{P}(V^H) \rightarrow \mathbb{R}_{\geq 0}$  and  $\phi_E : \mathcal{P}(E^G) \times \mathcal{P}(E^H) \rightarrow \mathbb{R}_{\geq 0}$  are functions that measures the dissimilarity between two sets of nodes and edges, respectively, and  $\alpha_0, \alpha_1, \alpha_2 \in \mathbb{R}_{\geq 0}$  are weights assigned to the dissimilarities between the nodes, the outer edges, and the inner edges. We now define  $d_{\mathcal{R}}$  as

$$d_{\mathcal{R}}(\mathcal{R}_L^G(u), \mathcal{R}_L^H(v)) = \sum_{l=0}^{L-1} \lambda_l d_{\mathcal{L}}(\mathcal{L}_l^G(u), \mathcal{L}_l^H(v)), \tag{1}$$

where  $\lambda_l \in \mathbb{R}_{\geq 0}$  are weights assigned to the distances between the layers.

Recall that we are defining  $d_{\mathcal{R}}$  to the purpose of populating a LSAP instance  $\mathbf{C}$  which is then used to derive an upper bound for GED. Since we want this upper bound to be as tight as possible, we want  $d_{\mathcal{R}}(\mathcal{R}_L^G(u), \mathcal{R}_L^H(v))$  to be small if and only if we have good reasons to assume that substituting  $u$  by  $v$  leads to a small overall edit cost. This can be achieved by defining the functions  $\phi_V$  and  $\phi_E$  in a way that makes crucial use of the edit cost functions  $c_V$  and  $c_E$ :

**LSAPE Based Definition of  $\phi_V$  and  $\phi_E$ .** Let  $U = \{u_1, \dots, u_r\} \subseteq V^G$  and  $V = \{v_1, \dots, v_s\} \subseteq V^H$  be two node sets. Then a LSAP instance  $\mathbf{C} = (c_{i,k}) \in \mathbb{R}^{(r+1) \times (s+1)}$  is defined by setting  $c_{i,k} = c_V(u_i, v_k)$ ,  $c_{i,s+1} = c_V(i, \epsilon)$ , and  $c_{r+1,k} = c_V(\epsilon, v_k)$  for all  $i \in \{1, \dots, r\}$  and all  $k \in \{1, \dots, s\}$ . This instance is solved—either optimally in  $O(\min\{r, s\}^2 \max\{r, s\})$  time or greedily in  $O(rs)$  time—and  $\phi_V$  is defined to return  $\mathbf{C}(\pi^*) / \max\{|U|, |V|, 1\}$ , where  $\mathbf{C}(\pi^*)$  is the cost of the computed solution  $\pi^*$ . We normalize by the sizes of  $U$  and  $V$  in order not to overrepresent large layers. The function  $\phi_E$  can be defined analogously.

**Multiset Intersection Based Definition of  $\phi_V$  and  $\phi_E$ .** Alternatively, we suggest to define  $\phi_V$  as

$$\phi_V(U, V) = \left[ \overline{c_V^{U,\epsilon}} \delta_{|U| \geq |V|} (|U| - |V|) + \overline{c_V^{\epsilon,V}} (1 - \delta_{|U| \geq |V|}) (|V| - |U|) + \overline{c_V^{U,V}} (\min\{|U|, |V|\} - |\ell_V^G[[U]] \cap \ell_V^H[[V]]|) \right] / \max\{|U|, |V|, 1\},$$

where  $\delta_{|U| \geq |V|}$  equals 1 if  $|U| \geq |V|$  and 0 otherwise,  $\overline{c_V^{U,\epsilon}}$ ,  $\overline{c_V^{\epsilon,V}}$ , and  $\overline{c_V^{U,V}}$  are the average costs of deleting a node in  $U$ , inserting a node in  $V$ , and substituting a node in  $U$  by a differently labeled node in  $V$ , and  $\ell_V^G[[U]]$  and  $\ell_V^H[[V]]$  are the multiset images of  $U$  and  $V$  under the labelling functions  $\ell_V^G$  and  $\ell_V^H$ . Again,  $\phi_E$  can be defined analogously. Note that, if the edit costs are quasimetric, then the LSAP based definition of  $\phi_V$  and  $\phi_E$  given above leads to the same number of node or edge substitutions, insertions, or deletions as the multiset intersection based definition; and if all substitution, insertion, and deletion costs are the same, then the two definitions are equivalent (cf. Proposition 1). Therefore, the

multiset intersection based approach for defining  $\phi_V$  and  $\phi_E$  can be seen as a proxy for the one based on LSAP. The advantage of using multiset intersection is that it allows for a very quick evaluation of  $\phi_V$  and  $\phi_E$ . In fact, since multiset intersections can be computed in quasilinear time [17], the dominant operation is the computation of the average substitution cost, which requires quadratic time. The drawback is that we lose some of the information encoded in the layers.

**Proposition 1.** *If all node substitution costs are equal to a constant  $c_V^S$ , all node removal costs to  $c_V^R$ , and all node insertion costs to  $c_V^I$  with  $c_V^S \leq c_V^R + c_V^I$ , then both definitions of  $\phi_V$  coincide. For  $\phi_E$ , an analogous proposition holds.*

*Proof.* We assume w. l. o. g. that  $|U| \leq |V|$ . Then, from  $c_V^S \leq c_V^R + c_V^I$  and by the first proposition in [5], the optimal solution  $\pi^*$  does not contain removals and contains exactly  $|V| - |U|$  insertions. The optimal cost  $\mathbf{C}(\pi^*)$  is thus reduced to the cost of  $|V| - |U|$  insertions plus  $c_V^S$  times the number of non identical substitutions. This last quantity is provided by  $\min\{|U|, |V|\} - l_V^G[[U]] \cap l_V^H[[V]]$ . We thus have:

$$\mathbf{C}(\pi^*) = c_V^I(|V| - |U|) + c_V^S(\min\{|U|, |V|\} - l_V^G[[U]] \cap l_V^H[[V]])$$

Since costs are constant, we have  $\overline{c_V^{\epsilon, \epsilon}} = c_V^R, \overline{c_V^{U, V}} = c_V^S$ , and  $\overline{c_V^{\epsilon, V}} = c_V^I$ , which provides the expected result. The proof for  $\phi_E$  is analogous.  $\square$

### 3.2 Algorithms and Choice of Meta-parameters

**Construction of the Rings and Overall Runtime Complexity.** Figure 2 shows how to build the rings via breadth-first search. Clearly, constructing all rings of a graph  $G$  requires  $O(|V^G|(|V^G| + |E^G|))$  time. After constructing the rings, the LSAP instance  $\mathbf{C}$  must be populated. Depending on the choice of  $\phi_V$  and  $\phi_E$ , this requires  $O(|\text{supp}(\lambda)||V^G||V^H|\Omega^3)$  or  $O(|\text{supp}(\lambda)||V^G||V^H|\Omega^2)$  time, where  $\Omega$  is the size of the largest set contained in one of the rings of  $G$  and  $H$ , and  $\text{supp}(\lambda)$  is the support of  $\lambda$ . Finally,  $\mathbf{C}$  is solved optimally in  $O(\min\{|V^G|, |V^H|\}^2 \max\{|V^G|, |V^H|\})$  time or greedily in  $O(|V^G||V^H|)$  time.

**Choice of the Meta-parameters  $\alpha$ ,  $\lambda$ , and  $L$ .** When introducing  $d_L$  and  $d_R$  in Sect. 3.1, we allowed  $\alpha$  and  $\lambda$  to be arbitrary vectors from  $\mathbb{R}_{\geq 0}^3$  and  $\mathbb{R}_{\geq 0}^L$ . However, we can be more restrictive: Since LSAP does not care about scaling, we can assume w. l. o. g. that  $\alpha$  and  $\lambda$  are simplex vectors, i. e., that we have  $\sum_{s=0}^2 \alpha_s = \sum_{l=0}^{L-1} \lambda_l = 1$ . This reduces the search space for  $\alpha$  and  $\lambda$  but still leaves us with too many degrees of freedom for choosing them via grid search. We hence suggest to learn  $\alpha$  and  $\lambda$  with the help of a blackbox optimizer [15]. For a training set of graphs  $\mathcal{T}$  and a fixed  $L \in \mathbb{N}_{>0}$ , the optimizer should minimize

$$obj(\alpha, \lambda) = \left[ \mu + (1 - \mu) \left( \frac{|\text{supp}(\lambda)| - 1}{\max\{1, L - 1\}} \right) \right] \sum_{(G, H) \in \mathcal{T}^2} \text{RING}_{\alpha, \lambda}^{\phi_V, \phi_E}(G, H)$$

and respect the constraints that  $\alpha$  and  $\lambda$  are simplex vectors.  $\text{RING}_{\alpha, \lambda}^{\phi_V, \phi_E}(G, H)$  is the upper bound for  $\text{GED}(G, H)$  returned by  $\text{RING}$  given fixed  $\alpha$ ,  $\lambda$ ,  $\phi_V$ , and

**Input:** A graph  $G$ , a node  $u \in V^G$ , and a constant  $L \in \mathbb{N}_{>0}$ .  
**Output:** The ring  $\mathcal{R}_L^G(u)$  rooted at  $u$ .

```

 $l \leftarrow 0$ ;  $V \leftarrow \emptyset$ ;  $OE \leftarrow \emptyset$ ;  $IE \leftarrow \emptyset$ ;  $\mathcal{R}_L^G(u) \leftarrow ((\emptyset, \emptyset, \emptyset))_{l=0}^{L-1}$ ; // initialize ring
 $d[u] \leftarrow 0$ ; for  $u' \in V^G \setminus \{u\}$  do  $d[u'] \leftarrow \infty$ ; // initialize distances to root
for  $e \in E^G$  do  $\text{discovered}[e] \leftarrow \text{false}$ ; // mark all edges as undiscovered
 $\text{open} \leftarrow \{u\}$ ; // initialize FIFO queue
while  $\text{open} \neq \emptyset$  do // main loop
   $u' \leftarrow \text{open.pop}()$ ; // pop node from queue
  if  $d[u'] > l$  then // the  $l^{\text{th}}$  layer is complete
     $\mathcal{R}_L^G(u)_l = (V, OE, IE)$ ;  $l \leftarrow l + 1$ ; // store  $l^{\text{th}}$  layer and increment  $l$ 
     $V \leftarrow \emptyset$ ;  $OE \leftarrow \emptyset$ ;  $IE \leftarrow \emptyset$ ; // reset nodes, inner, and outer edges
   $V \leftarrow V \cup \{u'\}$ ; //  $u'$  is node at  $l^{\text{th}}$  layer
  for  $u'u'' \in E^G$  do // iterate through neighbours of  $u'$ 
    if  $\text{discovered}[u'u'']$  then continue; // skip discovered edges
    if  $d[u''] = \infty$  then // found new node
       $d[u''] \leftarrow l + 1$ ; // set distance of new node
      if  $d[u''] < L$  then  $\text{open.push}(u'')$ ; // add close new node to queue
    if  $d[u''] = l$  then  $IE \leftarrow IE \cup \{u'u''\}$ ; //  $u'u''$  is inner edge at  $l^{\text{th}}$  layer
    else  $OE \leftarrow OE \cup \{u'u''\}$ ; //  $u'u''$  is outer edge at  $l^{\text{th}}$  layer
     $\text{discovered}[u'u''] \leftarrow \text{true}$ ; // mark  $u'u''$  as discovered
 $\mathcal{R}_L^G(u)_l = (V, OE, IE)$ ; return  $\mathcal{R}_L^G(u)$ ; // store last layer and return ring

```

**Fig. 2.** Construction of rings via Breadth-first search.

$\phi_E$ , and  $\mu \in [0, 1]$  is a tuning parameter that should be close to 1 if one wants to optimize for tightness and close to 0 if one wants to optimize for runtime. We include  $|\text{supp}(\boldsymbol{\lambda})| - 1$  in the objective, because if  $\boldsymbol{\lambda}$ 's support is small, only few layer distances have to be computed (cf. Eq. 1). In particular,  $|\text{supp}(\boldsymbol{\lambda})| = 1$  means that RING's runtime cannot be decreased any further via modification of  $\boldsymbol{\lambda}$ , which is why, in this case, the  $(1 - \mu)$ -part of the objective is set to 0.

Before building the rings for the graphs contained in the training set,  $L$  should be set to an upper bound for their diameters, e. g., to  $L = 1 + \max_{G \in \mathcal{T}} |V^G|$ . After the rings have been build,  $L$  can be lowered to  $L = 1 + \max\{l \mid \exists G \in \mathcal{T}, u \in V^G : \mathcal{R}_L^G(u)_l \neq (\emptyset, \emptyset, \emptyset)\} = 1 + \max_{G \in \mathcal{T}} \text{diam}(G)$  (cf. Remark 1). In the next step, the blackbox optimizer should be run, which returns an optimized pair of parameter vectors  $(\boldsymbol{\alpha}^*, \boldsymbol{\lambda}^*)$ . As the  $l^{\text{th}}$  layers contribute to  $d_{\mathcal{R}}$  only if  $l \in \text{supp}(\boldsymbol{\lambda}^*)$  (cf. Eq. 1),  $L$  can then be further lowered to  $L = 1 + \max_{l \in \text{supp}(\boldsymbol{\lambda}^*)} l$ .

## 4 Empirical Evaluation

We tested on the datasets MAO, PAH, ALKANE, and ACYCLIC, which contain graphs representing chemical compounds. For all datasets, we used the (non-uniform) edit costs 1 defined in [1]. We tested three variants of our method:

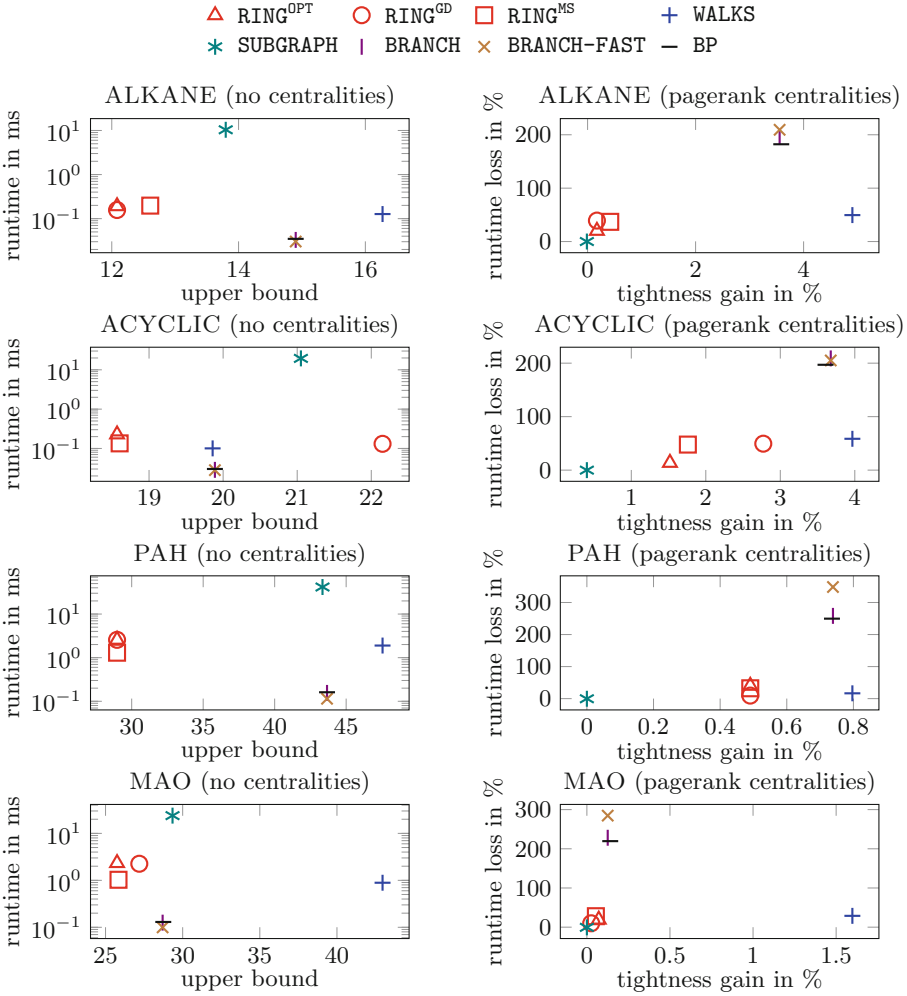


Fig. 3. Results of the experiments.

RING<sup>OPT</sup> uses optimal LSAPe for defining the distance functions  $\phi_V$  and  $\phi_E$ , RING<sup>GD</sup> uses greedy LSAPe, and RING<sup>MS</sup> uses the multiset intersection based approach. We compared them to instantiations of LSAPe-GED that can cope with non-uniform edit costs: BP, BRANCH, BRANCH-FAST, SUBGRAPH, and WALKS. As WALKS assumes that the costs of all edit operation types are constant, we slightly extended it by averaging the costs before each run. In order to handle the exponential complexity of SUBGRAPH, we enforced a time limit of 1 ms for computing a cell  $c_{i,k}$  of its LSAPe instance. All methods were run with and without pagerank centralities with the meta-parameter  $\beta$  set to 0.3, which, in [12], is reported to be the setting that yields the tightest average upper bound.

For learning the meta-parameters of  $\text{RING}^{\text{OPT}}$ ,  $\text{RING}^{\text{GD}}$ ,  $\text{RING}^{\text{MS}}$ ,  $\text{SUBGRAPH}$ , and  $\text{WALKS}$ , we picked a training set  $\mathcal{T} \subset \mathcal{D}$  with  $|\mathcal{T}| = 50$  for each dataset  $\mathcal{D}$ . As suggested in [6, 8], we learned the parameter  $L$  of the methods  $\text{SUBGRAPH}$  and  $\text{WALKS}$  by picking the  $L \in \{1, 2, 3, 4, 5\}$  which yielded the tightest average upper bound on  $\mathcal{T}$ . For choosing the meta-parameters of the variants of  $\text{RING}$ , we proceeded as suggested in Sect. 3.2: We set the tuning parameter  $\mu$  to 1 and used  $\text{NOMAD}$  [9] as our blackbox optimizer, which we initialized with 100 randomly constructed simplex vectors  $\alpha$  and  $\lambda$ . All methods are implemented in C++ and use the same implementation of the  $\text{LSAPE}$  solver proposed in [5]. Except for  $\text{WALKS}$ , all methods allow to populate the  $\text{LSAPE}$  instance  $\mathbf{C}$  in parallel and were set up to run in five threads. Tests were run on a machine with two Intel Xeon E5-2667 v3 processors with 8 cores each and 98 GB of main memory.<sup>1</sup>

For each dataset  $\mathcal{D}$ , we ran each method with and without pagerank centralities on each pair  $(G, H) \in \mathcal{D} \times \mathcal{D}$  with  $G \neq H$ . We recorded the runtime and the value of the returned upper bound for  $\text{GED}$ . Figure 3 shows the results of our experiments. The first column shows the average runtimes and upper bounds of the tested methods without centralities. The second column shows the effect of including centralities. On all datasets,  $\text{RING}^{\text{OPT}}$  yielded the tightest upper bound. Also  $\text{RING}^{\text{MS}}$  performed excellently, as its upper bound deviated from the one produced by  $\text{RING}^{\text{OPT}}$  by at most 4.15% (on  $\text{ALKANE}$ ). At the same time, on the datasets  $\text{ACYCLIC}$ ,  $\text{PAH}$ , and  $\text{MAO}$ ,  $\text{RING}^{\text{MS}}$  was around two times faster than  $\text{RING}^{\text{OPT}}$ . On the contrary,  $\text{RING}^{\text{GD}}$  was not significantly faster than  $\text{RING}^{\text{OPT}}$  and, on  $\text{ACYCLIC}$ , produced a 16.18% looser upper bound.

All competitors produced significantly looser upper bounds than our algorithms. In terms of runtime, our algorithms were outperformed by  $\text{BRANCH}$ ,  $\text{BRANCH-FAST}$ , and  $\text{BP}$ , performed similarly to  $\text{WALKS}$ , and were much faster than  $\text{SUBGRAPH}$ . Adding pagerank centralities did not improve the overall performance of the tested methods: It led to a maximal tightness gain of 4.90% ( $\text{WALKS}$  on  $\text{ALKANE}$ ) and dramatically increased the runtimes of some algorithms.

## 5 Conclusions and Future Work

In this paper, we have presented  $\text{RING}$ , a new instantiation of the paradigm  $\text{LSAPE-GED}$  which defines the local structure of a node  $u$  as a collection of node and edge sets at fixed distances from  $u$ . An empirical evaluation has shown that  $\text{RING}$  produces the tightest upper bound among all instantiations of  $\text{LSAPE-GED}$ . In the future, we will use ring structures for defining feature vectors of node assignments to be used in a machine learning based approach for approximating  $\text{GED}$ . Furthermore, we will examine how using  $\text{RING}$  for initialization affects the performance of the local search methods suggested in [4, 7, 11, 14].

<sup>1</sup> Source code and datasets: <http://www.inf.unibz.it/~blumenthal/gedlib.html>.

## References

1. Abu-Aisheh, Z., Gaüzere, B., Bougleux, S., Ramel, J.Y., Brun, L., Raveaux, R., Héroux, P., Adam, S.: Graph edit distance contest 2016: results and future challenges. *Pattern Recogn. Lett.* **100**, 96–103 (2017). <https://doi.org/10.1016/j.patrec.2017.10.007>
2. Blumenthal, D.B., Gamper, J.: Improved lower bounds for graph edit distance. *IEEE Trans. Knowl. Data Eng.* **30**(3), 503–516 (2018). <https://doi.org/10.1109/TKDE.2017.2772243>
3. Blumenthal, D.B., Gamper, J.: On the exact computation of the graph edit distance. *Pattern Recogn. Lett.* (2018). <https://doi.org/10.1016/j.patrec.2018.05.002>
4. Bougleux, S., Brun, L., Carletti, V., Foggia, P., Gaüzère, B., Vento, M.: Graph edit distance as a quadratic assignment problem. *Pattern Recogn. Lett.* **87**, 38–46 (2017). <https://doi.org/10.1016/j.patrec.2016.10.001>
5. Bougleux, S., Gaüzère, B., Blumenthal, D.B., Brun, L.: Fast linear sum assignment with error-correction and no cost constraints. *Pattern Recogn. Lett.* (2018). <https://doi.org/10.1016/j.patrec.2018.03.032>
6. Carletti, V., Gaüzère, B., Brun, L., Vento, M.: Approximate graph edit distance computation combining bipartite matching and exact neighborhood substructure distance. In: Liu, C.-L., Luo, B., Kropatsch, W.G., Cheng, J. (eds.) *GbRPR 2015*. LNCS, vol. 9069, pp. 188–197. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-18224-7\\_19](https://doi.org/10.1007/978-3-319-18224-7_19)
7. Ferrer, M., Serratos, F., Riesen, K.: A first step towards exact graph edit distance using bipartite graph matching. In: Liu, C.-L., Luo, B., Kropatsch, W.G., Cheng, J. (eds.) *GbRPR 2015*. LNCS, vol. 9069, pp. 77–86. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-18224-7\\_8](https://doi.org/10.1007/978-3-319-18224-7_8)
8. Gaüzère, B., Bougleux, S., Riesen, K., Brun, L.: Approximate graph edit distance guided by bipartite matching of bags of walks. In: Fränti, P., Brown, G., Loog, M., Escolano, F., Pelillo, M. (eds.) *S+SSPR 2014*. LNCS, vol. 8621, pp. 73–82. Springer, Heidelberg (2014). [https://doi.org/10.1007/978-3-662-44415-3\\_8](https://doi.org/10.1007/978-3-662-44415-3_8)
9. Le Digabel, S.: Algorithm 909: NOMAD: nonlinear optimization with the MADS algorithm. *ACM Trans. Math. Softw.* **37**(4), 44:1–44:15 (2011). <https://doi.org/10.1145/1916461.1916468>
10. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**(7), 950–959 (2009). <https://doi.org/10.1016/j.imavis.2008.04.004>
11. Riesen, K., Bunke, H.: Improving bipartite graph edit distance approximation using various search strategies. *Pattern Recogn.* **48**(4), 1349–1363 (2015). <https://doi.org/10.1016/j.patcog.2014.11.002>
12. Riesen, K., Bunke, H., Fischer, A.: Improving graph edit distance approximation by centrality measures. In: *ICPR 2014*, pp. 3910–3914. IEEE Computer Society (2014). <https://doi.org/10.1109/ICPR.2014.671>
13. Riesen, K., Ferrer, M., Fischer, A., Bunke, H.: Approximation of graph edit distance in quadratic time. In: Liu, C.-L., Luo, B., Kropatsch, W.G., Cheng, J. (eds.) *GbRPR 2015*. LNCS, vol. 9069, pp. 3–12. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-18224-7\\_1](https://doi.org/10.1007/978-3-319-18224-7_1)
14. Riesen, K., Fischer, A., Bunke, H.: Improved graph edit distance approximation with simulated annealing. In: Foggia, P., Liu, C.-L., Vento, M. (eds.) *GbRPR 2017*. LNCS, vol. 10310, pp. 222–231. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-58961-9\\_20](https://doi.org/10.1007/978-3-319-58961-9_20)



15. Rios, L.M., Sahinidis, N.V.: Derivative-free optimization: a review of algorithms and comparison of software implementations. *J. Global Optim.* **56**(3), 1247–1293 (2013). <https://doi.org/10.1007/s10898-012-9951-y>
16. Serratos, F., Cortés, X.: Graph edit distance: moving from global to local structure to solve the graph-matching problem. *Pattern Recogn. Lett.* **65**, 204–210 (2015). <https://doi.org/10.1016/j.patrec.2015.08.003>
17. Zeng, Z., Tung, A.K.H., Wang, J., Feng, J., Zhou, L.: Comparing stars: on approximating graph edit distance. *PVLDB* **2**(1), 25–36 (2009). <https://doi.org/10.14778/1687627.1687631>
18. Zheng, W., Zou, L., Lian, X., Wang, D., Zhao, D.: Efficient graph similarity search over large graph databases. *IEEE Trans. Knowl. Data Eng.* **27**(4), 964–978 (2015). <https://doi.org/10.1109/TKDE.2014.2349924>



# Graph Edit Distance in the Exact Context

Mostafa Darwiche<sup>1,2(✉)</sup>, Romain Raveaux<sup>1</sup>, Donatello Conte<sup>1</sup>,  
and Vincent T'Kindt<sup>2</sup>

<sup>1</sup> Université de Tours, LIFAT EA6300, 64 Avenue Jean Portalis,  
37200 Tours, France  
{mostafa.darwiche,romain.raveaux,donatello.conte}@univ-tours.fr  
<sup>2</sup> Université de Tours, LIFAT EA6300, ROOT ERL CNRS 7002,  
64 Avenue Jean Portalis, 37200 Tours, France  
tkindt@univ-tours.fr

**Abstract.** This paper presents a new Mixed Integer Linear Program (MILP) formulation for the Graph Edit Distance (GED) problem. The contribution is an exact method that solves the GED problem for attributed graphs. It has an advantage over the best existing one when dealing with the case of dense of graphs, because all its constraints are independent from the number of edges in the graphs. The experiments have shown the efficiency of the new formulation in the exact context.

**Keywords:** Graph Edit Distance · Graph Matching  
Mixed Integer Linear Program

## 1 Introduction

Graphs are very powerful in modeling structural relations of objects and patterns. A graph consists of two sets of vertices and edges. The vertices represent the main components, while the edges show the link between those components. In a graph, it is also possible to store information and features about the object, by assigning attributes to vertices and edges. Graphs have been used in many applications and fields, such as *Pattern Recognition* to model objects in images and videos [13]. Also, graphs form a natural representation of the atom-bond structure of molecules, therefore they have applications in *Cheminformatics* field [11]. A common task is then, the ability to compare graphs or find (dis)similarities between them. Such a task enables comparing objects and patterns that are represented by graphs, and this is known as *Graph Matching* (GM). GM has been split into different sub-problems, which mainly fall under two categories: *exact* and *error tolerant*. The first one is very strict, while the second is more flexible and tolerant to differences in topologies and attributes, which makes it more suitable for real-life scenarios.

*Graph Edit Distance* (GED) problem is an error-tolerant graph matching problem. It provides a dissimilarity measure between two graphs, by computing

the cost of editing one graph to transform it into another. The set of edit operations are substitution, insertion and deletion, and can be applied on both vertices and edges. There is a cost associated to each edit operation. Solving the GED problem consists in finding the sequence of edit operations that minimizes the total cost. GED, by concept, is known to be flexible because it has been shown that changing the edit cost properties can result in solving other matching problems such as, maximum common subgraph, graph and subgraph isomorphism [4]. GED is a minimization problem that was proven to be NP-hard. The problem is complex and hence it was mostly treated by heuristic methods in order to compute sub-optimal solutions in reasonable time. A famous heuristic is called *Bipartite Graph Matching* (BP), which is known to be fast [12]. BP breaks down the GED problem into a linear sum assignment problem that can be solved in polynomial time, using the *Hungarian* algorithm [10]. BP was integrated later in other heuristics such as *Fast BP*, *Square BP* and *Beam-search BP* [6,14]. Two new heuristics: *Integer Projected Fixed Point* (IPFP) and *Graduate Non Convexity and Concavity Procedure* (GNCCP), were proposed by Bougleux et al. [3]. Both are adapted to operate over a *Quadratic Assignment Problem* (QAP) that models the GED. These heuristics aim at approximating the quadratic objective function to compute a solution and then improve it by applying projection methods. In a recent work by Darwiche et al. [5], a heuristic called *Local Branching GED* was proposed, that is based on local searches in the solution space of a *Mixed Integer Linear Program* (MILP). On the other hand, and in the exact context (e.g. methods that compute optimal solutions), there are three MILP formulations in the literature. Only two of them are designed to solve the general GED problem [8]. The third formulation was designed by Justice and Hero [7], and it is the most efficient formulation. However, it only deals with a special case of the GED problem, where attributes on edges are ignored and a constant cost is assigned to edges edit operations. As well, in the exact context, there is a branch and bound algorithm [2], which was shown later to be less efficient than MILP formulations.

The present work is with the interest of designing a new MILP formulation to solve the GED problem, and so contributes to the exact methods for GED. A new efficient formulation is proposed that has good performance w.r.t. existing formulations in the literature. The new formulation is inspired by *F2*, which is proposed by Lerouge et al. [8]. It is an improvement to *F2* by modifying the variables and the constraints. It has the advantage over *F2*, that the constraints are independent from the number of edges in the graphs. The remainder is organized as follows: Sect. 2 presents the definition of the GED problem, followed with a review of *F2* formulation. Then, Sect. 3 details the improved formulation. Section 4 shows the results of the computational experiments. Finally, Sect. 5 highlights some concluding remarks.

## 2 GED Definition and F2 Formulation

### 2.1 GED Problem Definition

An attributed graph is a 4-tuple  $G = (V, E, \mu, \xi)$  where,  $V$  is the set of vertices,  $E$  is the set of edges, such that  $E \subseteq V \times V$ ,  $\mu : V \rightarrow L_V$  (resp.  $\xi : E \rightarrow L_E$ ) is the function that assigns attributes to a vertex (resp. an edge), and  $L_V$  (resp.  $L_E$ ) is the label space for vertices (resp. edges).

Next, given two graphs  $G = (V, E, \mu, \xi)$  and  $G' = (V', E', \mu', \xi')$ , GED is the task of transforming one graph source into another graph target. To accomplish this, GED introduces the vertices and edges edit operations:  $(i \rightarrow k)$  is the substitution of two vertices,  $(i \rightarrow \epsilon)$  is the deletion of a vertex, and  $(\epsilon \rightarrow k)$  is the insertion of a vertex, with  $i \in V, k \in V'$  and  $\epsilon$  refers to the empty node. The same logic goes for edges. The set of operations that reflects a valid transformation of  $G$  into  $G'$  is called a complete edit path, defined as  $\lambda(G, G') = \{o_1, \dots, o_k\}$ , where  $o_i$  is an elementary vertex (or edge) edit operation and  $k$  is the number of operations. GED is then

$$d_{min}(G, G') = \min_{\lambda \in \Gamma(G, G')} \sum_{o_i \in \lambda} \ell(o_i) \tag{1}$$

where  $\Gamma(G, G')$  is the set of all complete edit paths,  $d_{min}$  represents the minimal cost obtained by a complete edit path  $\lambda(G, G')$ , and  $\ell(\cdot)$  is the cost function that assigns costs to elementary edit operations.

### 2.2 Mixed Integer Linear Program

The general MILP formulation is of the form:

$$\min_x c^T x \tag{2}$$

$$Ax \geq b \tag{3}$$

$$x_i \in \{0, 1\}, \forall i \in B \tag{4}$$

$$x_j \in \mathbb{N}, \forall j \in I \tag{5}$$

$$x_k \in \mathbb{R}, \forall k \in C \tag{6}$$

where  $c \in \mathbb{R}^n$  and  $b \in \mathbb{R}^m$  are vectors of coefficients,  $A \in \mathbb{R}^{m \times n}$  is a matrix of coefficients.  $x$  is a vector of variables to be computed. The variable index set is split into three sets  $(B, I, C)$ , respectively stands for binary, integer and continuous. This formulation minimizes an objective function (Eq. 2) w.r.t. a set of linear inequality constraints (Eq. 3) and the bounds imposed on variables  $x$  e.g. integer or binary. A feasible solution to this formulation is a vector  $x$  with the proper values based on their defined types, that satisfies all the constraints. The optimal solution is a feasible solution that has the minimum objective function value. This approach of modeling decision problems (i.e. problems with binary and integer variables) is very efficient, especially for hard optimization problems.

### 2.3 F2 Formulation

F2 is the best MILP formulation for the GED problem in the literature, it was proposed by Lerouge et al. [8]. It is based on a previous and straightforward MILP formulation, referred to as F1, by the same authors. F2 formulation is a more compact and improved version of F1 by reducing the number of variables and constraints. The compactness of F2 comes from the design of the objective function to be optimized. At first, it considers all vertices and edges of  $G$  as deleted and vertices and edges of  $G'$  as inserted. Then, it solves the problem of finding the cheapest assignments/matching between the two sets of vertices and the two sets of edges. The matching in this context is the substitution edit operations for vertices and edges. Once, the cheapest matching is computed, the deletion and insertion operations can be concluded. All the remaining vertices in  $V$  (resp. in  $V'$ ) that are not matched with any vertex in  $V'$  (resp. in  $V$ ), are considered as deleted (resp. inserted). The edges are treated in the same manner. Such design is helpful in reducing the number of variables and constraints in the formulation. In the following, F2 is detailed by defining the data of the problem, variables, objective function to minimize and constraints to respect.

*Data.* Given two graphs  $G = (V, E, \mu, \xi)$  and  $G' = (V', E', \mu', \xi')$ , the cost functions, in order to compute the cost of each vertex/edge edit operations, are known and defined. Therefore, vertices cost matrix  $[c_v]$  is computed as in Eq. 7 for every couple  $(i, k) \in V \times V'$ . The  $\epsilon$  column is added to store the cost of deletion  $i$  vertices, while the  $\epsilon$  row stores the costs of insertion  $k$  vertices. Following the same process, the matrix  $[c_e]$  is computed for every  $((i, j), (k, l)) \in E \times E'$ , plus the row/column  $\epsilon$  for deletion and insertion of edges.

$$c_v = \begin{matrix} & v_1 & v_2 & \dots & v_{|V'|} & \epsilon \\ \begin{bmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,|V'|} & c_{1,\epsilon} \\ c_{2,1} & c_{2,2} & \dots & c_{2,|V'|} & c_{2,\epsilon} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ c_{|V|,1} & c_{|V|,2} & \dots & c_{|V|,|V'|} & c_{|V|,\epsilon} \\ c_{\epsilon,1} & c_{\epsilon,2} & \dots & c_{\epsilon,|V|} & 0 \end{bmatrix} & \begin{matrix} u_1 \\ u_2 \\ \vdots \\ u_{|V|} \\ \epsilon \end{matrix} \end{matrix} \tag{7}$$

*Variables.* As mentioned earlier, F2 formulation focuses on finding the correspondences between the two sets of vertices and the two sets of edges. That is why two sets of decision variables are needed.

- $x_{i,k} \in \{0, 1\} \forall i \in V, \forall k \in V'$ ;  $x_{i,k} = 1$  when vertices  $i$  and  $k$  are matched, and 0 otherwise.
- $y_{ij,kl} \in \{0, 1\} \forall (i, j) \in E, \forall (k, l) \in E'$ ;  $y_{ij,kl} = 1$  when edge  $(i, j)$  is matched with  $(k, l)$ , and 0 otherwise.

*Objective Function.* The objective function to minimize is the following.

$$\begin{aligned} & \min_{x,y} \sum_{i \in V} \sum_{k \in V'} (c_v(i, k) - c_v(i, \epsilon) - c_v(\epsilon, k)) \cdot x_{i,k} \\ & + \sum_{(i,j) \in E} \sum_{(k,l) \in E'} (c_e(ij, kl) - c_e(ij, \epsilon) - c_e(\epsilon, kl)) \cdot y_{ij,kl} + \gamma \end{aligned} \tag{8}$$

The objective function minimizes the cost of assigning vertices and edges with the cost of substitution subtracting the cost of insertion and deletion. The  $\gamma$ , which is a constant and given in Eq. 9, compensates the subtracted costs of the assigned vertices and edges. This constant does not impact the optimization algorithm and it could be removed. It is there to obtain the GED value.

$$\gamma = \sum_{i \in V} c_v(i, \epsilon) + \sum_{k \in V'} c_v(\epsilon, k) + \sum_{(i,j) \in E} c_e(ij, \epsilon) + \sum_{(k,l) \in E'} c_e(\epsilon, kl) \tag{9}$$

*Constraints.*  $F2$  has 3 sets of constraints.

$$\sum_{k \in V'} x_{i,k} \leq 1 \quad \forall i \in V \tag{10}$$

$$\sum_{i \in V} x_{i,k} \leq 1 \quad \forall k \in V' \tag{11}$$

$$\sum_{(k,l) \in E'} y_{ij,kl} \leq x_{i,k} + x_{j,k} \quad \forall k \in V', \forall (i, j) \in E \tag{12}$$

Constraints 10 and 11 are to make sure that a vertex can be only matched with maximum one vertex. It is possible that a vertex is not assigned to any other, in this case it is considered as deleted or inserted. Here is the key point of this formulation:  $F2$  is flexible by allowing some vertices/edges not to be matched. The objective function gets to decide whether a substitution is cheaper than a deletion/insertion or not.  $\gamma$  takes care of the unmatched vertices/edges and includes their deletion or insertion costs to the objective function. Finally, constraints 12 guarantee preserving edges matching between two couple of vertices. In other words, to match two edges  $(i, j) \rightarrow (k, l)$ , their vertices must be matched first, i.e.  $i \rightarrow k$  and  $j \rightarrow l$  OR  $i \rightarrow l$  and  $j \rightarrow k$ .

The presented version of  $F2$  formulation, and for the sake of simplicity, is applied to undirected graphs. For the directed case, it simply splits the constraints 12 into two sets of constraints. For more details, please refer to the paper [8].

### 3 Improved MILP Formulation ( $F3$ )

#### 3.1 $F3$ Formulation

$F3$  is a new and an improved MILP formulation, inspired by  $F2$ , to solve the GED problem. It shares some parts of  $F2$  and it is defined as follows.

*Data.* Same as in  $F2$  formulation,  $F3$  uses the cost matrices  $[c_v]$  and  $[c_e]$ .

*Variables.*  $F3$  introduces two sets of decision variables  $x_{i,k}$  and  $y_{ij,kl}$  as in  $F2$ . However, it includes more  $y$  variables, by creating two variables:  $y_{ij,kl}$  and  $y_{ij,lk}$  for every  $((i, j), (k, l)) \in E \times E'$ . Let  $\bar{E}' = \{(l, k) : \forall (k, l) \in E'\}$ . The variables of the formulation are as follows.

- $x_{i,k} \in \{0, 1\} \forall i \in V, \forall k \in V'$ ;  $x_{i,k} = 1$  when vertices  $i$  and  $k$  are matched, and 0 otherwise.
- $y_{ij,kl} \in \{0, 1\} \forall (i, j) \in E, \forall (k, l) \in E' \cup \bar{E}'$ ;  $y_{ij,kl} = 1$  when edge  $(i, j)$  is matched with  $(k, l)$ , and 0 otherwise.

*Objective Function.* It is basically the same function as in  $F2$  formulation, except for the cost sum over the  $y$  variables to include all of them.

$$\begin{aligned} & \min_{x,y} \sum_{i \in V} \sum_{k \in V'} (c_v(i, k) - c_v(i, \epsilon) - c_v(\epsilon, k)) \cdot x_{i,k} \quad (8-a) \\ & + \sum_{(i,j) \in E} \sum_{(k,l) \in E' \cup \bar{E}'} (c_e(ij, kl) - c_e(ij, \epsilon) - c_e(\epsilon, kl)) \cdot y_{ij,kl} + \gamma \end{aligned}$$

*Constraints.*  $F3$  formulation shares the same sets of constraints 10 and 11, that assure a vertex is only matched with one vertex at most. However, it re-writes the constraints 12 in a different fashion.

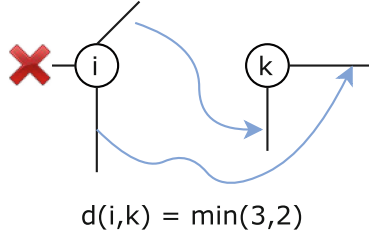
$$\sum_{(i,j) \in E} \sum_{(k,l) \in E' \cup \bar{E}'} y_{ij,kl} \leq d_{i,k} \times x_{i,k} \quad \forall i \in V, \forall k \in V' \quad (12-a)$$

With  $d_{i,k} = \min(\text{degree}(i), \text{degree}(k))$ . The degree of a vertex is the number of edges incident to the vertex. The constraints stands for: whenever two vertices are matched, e.g.  $(i \rightarrow k)$ , the maximum number of edges substitution that can be done is equal to the minimum degree of the two vertices. Figure 1 shows an example of the case. Two edges at most can be substituted and the third of  $i$  has to be deleted. Of course, the deletion of all edges is possible, if it costs less than the substitutions. These constraints force matching the edges and respecting the topological constraint defined in the GED problem.

The given formulation handles the case of undirected graphs. Though, it can be adapted to deal with the directed case, by setting  $\bar{E}' = \{\phi\}$  (because edges  $(i, j)$  are different from  $(j, i)$  and they are already included in  $E$ ), and replacing the objective function Eq. 8-a by the objective function of  $F2$  Eq. 8.

### 3.2 F2 vs. F3

The most important improvement in the proposed formulation is that  $F3$  has sets of constraints independent of the number of edges in the graphs. Constraints 10 and 11 are shared by both formulations and they do not include edges. However, constraints 12 rely on the edges of  $G$ , which is not the case of the constraints



**Fig. 1.** Example of edges assignment when assigning two vertices

12-a in  $F3$ . Table 1 shows the number of variables and constraints in both formulations. Clearly,  $F3$  has (2 times) more  $y$  variables than  $F2$ . The reason behind creating two  $y$  variables for each couple of edges, is to accommodate to the symmetry case that appears when dealing with undirected graphs, i.e.  $(i, j) = (j, i)$ . By doing so, the constraints 12 can be re-written differently by relying only on the vertices of the graphs (constraints 12-a). Note that, this comparison is done for undirected graphs. In the other case, the symmetry is discarded, and both formulations have the same number of variables.

**Table 1.** Nb. of variables and constraints in  $F2$  and  $F3$

	Nb. of variables	Nb. of constraints
$F2$	$ V  \times  V'  +  E  \times  E' $	$ V  +  V'  +  V  \times  E $
$F3$	$ V  \times  V'  +  E  \times  E'  \times 2$	$ V  +  V'  +  V  \times  V' $

In the GED problem, edge operations are driven by vertex-vertex matching. On this basis, the difficulty in  $F2$  and  $F3$  comes from the  $x$  decision variables, rather than the  $y$  variables. Moreover,  $F2$  formulation is more sensitive to the density of the graphs (% connectivity,  $D = \frac{2|E|}{|V|(|V|-1)}$ ), because its constraints depend on the edges, which is not the case in  $F3$ . This reasoning led to make the following two assumptions, by distinguishing between two cases:

1. Non-dense graphs: even if  $F3$  has more  $y$  variables than in  $F2$ , its performance will not be degraded compared to  $F2$ .
2. Dense graphs:  $F3$  will have less constraints than  $F2$ , since  $F3$  has a number of constraints independent from the number of edges. Consequently,  $F3$  tends to perform better than  $F2$ .

To validate those assumptions, both formulations are tested over two graph databases. The results are discussed in the next section.

## 4 Computational Experiment

### 4.1 Databases

Two databases are selected from the literature in order to evaluate  $F3$ .



*MUTA*. This database consists of graph that model chemical molecules [1]. It is commonly used when testing GED methods, mainly because it contains different subsets of small and large graphs. It allows exploiting GED methods and shows their behaviors when the instances get more difficult. There are 7 subsets, each of which has 10 graphs of same size (10 to 70 vertices) and a subset of also 10 graphs with mixed graph sizes. Each pair of graphs is considered as an instance. Therefore, a total of 800 instances (100 per subset) are considered in this experiment. The density of the graphs is very low ( $D = 7\%$ ), hence they are considered as non-dense graphs. The choice of the edit operations costs is based on the values defined in [1].

*CMUHOUSE*. This database contains 111 graphs corresponding to 3-D images of houses [9], each graph consists of 30 vertices with attributes described using Shape Context feature vector. The graphs are extracted from 3-D house images, where the houses are rotated with different angles. This is interesting because it enables testing and comparing graphs that represent the same house but positioned differently inside the images. For this database, there are 660 instances in total. The density of these graphs is higher than *MUTA* graphs,  $D = 18\%$ . Two versions of this database are considered: *CMUHOUSE-NA* is the version where attributes are not considered when calculating the costs; *CMUHOUSE-A* a second version with costs computed based on the functions given in [15].

## 4.2 Experiment Settings

Both formulations are implemented in C language, and solved by CPLEX 12.7.1 with time limit 900s. The tests were executed on a machine with the following configuration: Windows 7 (64-bit), Intel Xeon E5 4 cores and 8 GB RAM. For each formulation, the following values are computed for each subset of graphs:  $t_{avg}$  is the average CPU time in seconds for all instances,  $d_{avg}$  is the deviation percentage between the solutions obtained by one formulation, and the best computed by both formulations. For example, given an instance  $I$ , the deviation percentage for  $F3$  is equal to  $\frac{sol_I^{F3} - best_I}{best_I} \times 100$ , with  $best_I = \min(sol_I^{F2}, sol_I^{F3})$ . Lastly,  $\eta_I$  and  $\eta'_I$  represent, respectively, the number of optimal solutions obtained by a formulation, and the number of solutions for which, a given formulation has provided the minimum (smaller objective function value, without necessarily a proof of optimality).

## 4.3 Results and Analysis

*MUTA Results*. Table 2 shows the results obtained for both formulations for each subset of graphs. Looking at  $d_{avg}$  for  $F2$ , it scores the smallest values for all the subsets, except for subset 70. However, the gap between both formulations is small, especially with small instances (0% for subsets 10 and 20). In terms optimal solutions ( $\eta$ ),  $F3$  has higher numbers for subsets 30, 40, 50 and *Mixed*, with greater differences: for subsets 30 at 76 optimal solutions against 48, and subset

50 at 31 optimal solutions against 19. Regarding  $\eta'$ ,  $F2$  has higher numbers for most of the subsets (30, 50, 60 and *Mixed*). However,  $\eta'$  of  $F3$  are not far the ones of  $F2$ . At last,  $F2$  is faster than  $F3$  for small and medium subsets (10, 20, 30 and *Mixed*). But, for the rest of the subsets, both formulations suffer from high computation time and reach the time limit set (900s). The conclusion of this experiment: both formulations seems to be very close in terms of performance and efficiency in computing optimal solutions. It is hard to tell which formulation is better. This result corroborates the first assumption, that is  $F3$  is as good as  $F2$  in the case of non-dense graphs.

**Table 2.** Results of MUTA instances

		10	20	30	40	50	60	70	Mixed
F3	$t_{avg}(s)$	0.10	3.07	365.44	575.65	770.61	810.51	811.10	410.08
	$d_{avg}$	<b>0.00</b>	<b>0.00</b>	0.74	0.54	1.78	3.60	<b>2.55</b>	0.80
	$\eta$	<b>100</b>	<b>100</b>	<b>81</b>	<b>76</b>	<b>31</b>	10	10	<b>62</b>
	$\eta'$	<b>100</b>	<b>100</b>	91	<b>90</b>	68	53	<b>61</b>	78
F2	$t_{avg}(s)$	<b>0.05</b>	<b>0.99</b>	<b>320.35</b>	<b>571.65</b>	<b>766.63</b>	<b>802.94</b>	<b>802.69</b>	<b>370.36</b>
	$d_{avg}$	<b>0.00</b>	<b>0.00</b>	<b>0.21</b>	<b>0.51</b>	<b>1.52</b>	<b>1.46</b>	2.76	<b>0.15</b>
	$\eta$	<b>100</b>	<b>100</b>	79	48	19	<b>11</b>	<b>11</b>	61
	$\eta'$	<b>100</b>	<b>100</b>	<b>93</b>	84	<b>69</b>	<b>69</b>	60	<b>91</b>

**Table 3.** Results of CMUHOUSE instances

		CMUHOUSE-NA	CMUHOUSE-A
F3	$t_{avg}(s)$	<b>497.07</b>	416.75
	$d_{avg}$	<b>0.70</b>	<b>0.22</b>
	$\eta$	<b>365</b>	<b>633</b>
	$\eta'$	<b>644</b>	<b>652</b>
F2	$t_{avg}(s)$	880.74	<b>278.78</b>
	$d_{avg}$	604.11	4.68
	$\eta$	25	505
	$\eta'$	54	548

*CMUHOUSE Results.* Table 3 presents the results of both formulations for both versions of CMUHOUSE. In the case of CMUHOUSE-NA (no attributes), the instances seem to be harder than the version with attributes. When ignoring the attributes, the similarities between vertices and edges are high and it does not allow to easily differentiate between them. The average deviation for  $F3$  is 0.70% against 604.11% for  $F2$ , the difference is remarkably high. This is also seen when looking at  $\eta$  and  $\eta'$ , respectively, 365, 644 for  $F3$  against 25, 54 for  $F2$ .  $F3$  was

able to compute optimal solutions for more than 50% of the instances. It looks like  $F2$  had hard time with these instances in converging towards good solutions. The version with attributes (CMUHOUSE-A) is easier, but still  $F3$  has scored  $d_{avg} = 0.22\%$  against 4.68% for  $F2$ .  $F3$  has solved more instances to optimality (652) than  $F2$  (505). Based on these results, the second assumption also holds true. CMUHOUSE graphs are more dense than MUTA, which means that  $F3$  has less constraints, since all its constraints are independent from the number of edges in the graphs. As a result,  $F3$  has performed better than  $F2$ .

## 5 Conclusion

In this work, a new MILP formulation is proposed for the GED problem. The new formulation is an improvement to the best existing one. The results of the experiments have shown the efficiency of this formulation, especially in the case of dense graphs. This is due to the fact that, the constraints are independent from the edges in the graphs. The next step will be to evaluate the new formulation against more graph databases with different settings, i.e. graphs with high and very high densities.

## References

1. Abu-Aisheh, Z., Raveaux, R., Ramel, J.: A graph database repository and performance evaluation metrics for graph edit distance. In: Proceedings of Graph-Based Representations in Pattern Recognition - 10th IAPR-TC-15, pp. 138–147 (2015)
2. Abu-Aisheh, Z., Raveaux, R., Ramel, J.Y., Martineau, P.: An exact graph edit distance algorithm for solving pattern recognition problems. In: 4th International Conference on Pattern Recognition Applications and Methods 2015 (2015)
3. Bougleux, S., Brun, L., Carletti, V., Foggia, P., Gaüzère, B., Vento, M.: Graph edit distance as a quadratic assignment problem. *Pattern Recogn. Lett.* **87**, 38–46 (2017)
4. Bunke, H.: On a relation between graph edit distance and maximum common subgraph. *Pattern Recogn. Lett.* **18**(8), 689–694 (1997)
5. Darwiche, M., Conte, D., Raveaux, R., T'Kindt, V.: A local branching heuristic for solving a graph edit distance problem. *Comput. Oper. Res.* (2018). <https://doi.org/10.1016/j.cor.2018.02.002>. ISSN 0305-0548
6. Ferrer, M., Serratoso, F., Riesen, K.: Improving bipartite graph matching by assessing the assignment confidence. *Pattern Recogn. Lett.* **65**, 29–36 (2015)
7. Justice, D., Hero, A.: A binary linear programming formulation of the graph edit distance. *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(8), 1200–1214 (2006)
8. Lerouge, J., Abu-Aisheh, Z., Raveaux, R., Héroux, P., Adam, S.: New binary linear programming formulation to compute the graph edit distance. *Pattern Recogn.* **72**, 254–265 (2017). <https://doi.org/10.1016/j.patcog.2017.07.029>
9. Moreno-García, C.F., Cortés, X., Serratoso, F.: A graph repository for learning error-tolerant graph matching. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) S+SSPR 2016. LNCS, vol. 10029, pp. 519–529. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_46](https://doi.org/10.1007/978-3-319-49055-7_46)

10. Munkres, J.: Algorithms for the assignment and transportation problems. *J. Soc. Ind. Appl. Math.* **5**(1), 32–38 (1957)
11. Raymond, J.W., Willett, P.: Maximum common subgraph isomorphism algorithms for the matching of chemical structures. *J. Comput.-Aided Mol. Des.* **16**(7), 521–533 (2002)
12. Riesen, K., Neuhaus, M., Bunke, H.: Bipartite graph matching for computing the edit distance of graphs. In: Escolano, F., Vento, M. (eds.) *GbrPR 2007*. LNCS, vol. 4538, pp. 1–12. Springer, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-72903-7\\_1](https://doi.org/10.1007/978-3-540-72903-7_1)
13. Sanfeliu, A., Fu, K.S.: A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man Cybern. SMC* **13**(3), 353–362 (1983). <https://doi.org/10.1109/TSMC.1983.6313167>
14. Serratosa, F.: Computation of graph edit distance: reasoning about optimality and speed-up. *Image Vis. Comput.* **40**, 38–48 (2015)
15. Zhang, Z., Shi, Q., McAuley, J.J., Wei, W., Zhang, Y., Van Den Hengel, A.: Pair-wise matching through max-weight bipartite belief propagation. In: *CVPR*, vol. 5, p. 7 (2016)



# The VF3-Light Subgraph Isomorphism Algorithm: When Doing Less Is More Effective

Vincenzo Carletti<sup>(✉)</sup>, Pasquale Foggia<sup>(✉)</sup>, Antonio Greco, Alessia Saggese, and Mario Vento

Department of Information and Electrical Engineering and Applied Mathematics,  
University of Salerno, Fisciano, Italy  
{vcarletti,pfoggia,agreco,asaggese,mvento}@unisa.it

**Abstract.** We have recently introduced VF3, a general-purpose subgraph isomorphism algorithm that has demonstrated to be very effective on several datasets, especially on very large and very dense graphs.

In this paper we show that on some classes of graphs, the whole power of VF3 may become overkill; indeed, by removing some of the heuristics used in it, and as a consequence also some of the data structures that are required by them, we obtain an algorithm that is actually faster.

In order to provide a characterization of this modified algorithm, called VF3-Light, we have performed an evaluation using several kinds of graphs; besides comparing VF3-Light with VF3, we have also compared it to RI, a fast recent algorithm that is based on a similar approach.

## 1 Introduction

Graphs are a popular representation in *Structural Pattern Recognition*, where the object of interest can be decomposed into parts (represented as *nodes*) and significant information is attached to the relationships between parts (represented as *edges*). Applications where this kind of representation have been profitably used include computer vision, chemistry, biology, social network analysis, databases.

A common task on such representations is finding suitable correspondances between the structures of two graphs (*graph matching*); an important special case is the search for occurrences of a smaller graph (called *pattern*) inside a larger graph (called *target*). *Subgraph isomorphism* is a possible formulation of this problem, that has been widely investigated in the literature: see [1–3] for extensive reviews on subgraph isomorphism and other graph matching algorithms in the field of Pattern Recognition.

Many subgraph isomorphism algorithms (e.g. Ullmann’s [4], VF2 [5], L2G [6], RI/RI-DS [7]) are based on *Tree Search*. In this approach, the *search space* (also called *state space*) is conceptually defined as a tree of *states*, where each state correspond to a partial mapping of the pattern nodes onto target nodes. The root of the tree is the state corresponding to an empty mapping, while a new state is obtained from an existing one by adding to the mapping a pair

(pattern node, target node) that ensures the preservation of the structural constraints imposed by problem formulation. Algorithms based on this approach perform a depth-first visit of the state space with backtracking, in order to avoid the explicit construction of the whole state space. The algorithms essentially differ from each other in the order they visit search space, the heuristics they adopt for pruning unfruitful portions of the space, and the data structures they need to keep and update during the visit process; these factors, although they do not change the asymptotic worst case complexity (the problem is NP-complete), may greatly affect the actual execution times on graphs commonly found in applications.

The choice of the heuristics is often subject to a trade-off: a given heuristic may allow the algorithm to detect in advance that a candidate state is a dead end, saving the need to explore its successors. However, the time for evaluating this heuristic must be added to the time spent on each state. Furthermore, sophisticated heuristics usually need additional data structures to be kept during the visit process, and the contents of these structures have to be updated for each examined state, adding more time and in some cases more space to the requirements of the algorithm.

In [8] the authors have presented VF3, a recent algorithm based on this approach, especially devised to be effective on large and dense graphs, which are often problematic for other matching algorithms. VF3 is defined as an extension of a previous algorithm, named VF2. The authors demonstrate, using an extensive experimentation, that this algorithm is not only significantly faster than the original VF2, but also faster than other recent state-of-the-art algorithms.

In this paper, we introduce a simplified version of VF3, named VF3-Light, that avoids some of the heuristics used in VF3 and in its predecessor VF2. While the removal of these heuristics imply that the new algorithm has a reduced pruning ability, and thus may visit more states than VF3, VF3-Light can avoid keeping and updating some of the data structures needed by its predecessor. This in turn makes the visit of each state faster, and on some kinds of graphs the time saving is such to obtain a smaller overall matching time.

As we will show in the experimental section, a preliminary experimentation has demonstrated that this is indeed the case on several kinds of graphs, while on other types of graphs the full power of the complete VF3 heuristics still proves to be able to achieve the fastest results.

## 2 The Proposed Method

In this section, we will first present a short description of the original VF3 algorithm (the reader is referred to [8] for more details). Then we will discuss the heuristics that have been removed to obtain VF3-Light, highlighting the impact on the data structures that the algorithm needs maintain.

We will denote as  $G = (V, E)$  a graph with the set of its nodes  $V$  and the set of its edges  $E \subset V \times V$ . The pattern (smaller) graph will be  $G_1 = (V_1, E_1)$ , and the target (larger) graph will be  $G_2 = (V_2, E_2)$ . Nodes and edges usually

have also *labels* or *attributes*, that are represented using two labeling functions:  $\lambda_v : V_1 \cup V_2 \rightarrow L_v$  for the nodes, and  $\lambda_e : E_1 \cup E_2 \rightarrow L_e$  for the edges. Given a node  $u \in V_1$ , we will denote as  $\mathcal{S}_1(u)$  the set of all the successors of  $u$ , i.e. the nodes reached by an edge starting from  $u$ , and as  $\mathcal{P}_1(u)$  the predecessors, i.e. the starting nodes of edges arriving to  $u$ . We similarly define  $\mathcal{S}_2(v)$  and  $\mathcal{P}_2(v)$  for  $v \in V_2$ . Graph matching is the problem of finding a mapping function  $M : V_1 \rightarrow V_2$  satisfying some structural constraints. For subgraph isomorphism [1], the constraints are that  $M$  is injective and *structure preserving*, i.e. the nodes put in correspondance must have the same structure considering both the presence and the absence of edges.

### 2.1 Overview of the VF3 Algorithm

Before describing the algorithm, let us introduce some notations that will be used in the following. As previously said, the algorithm visits a search space that is conceptually organized as a tree of states, with each state  $s$  representing a partial mapping built so far by the algorithm. In this tree two states are connected if the second can be obtained from the first by adding a pair of nodes  $(u, v) \in V_1 \times V_2$  to its partial mapping.

```

function VF3( $G_1, G_2$ )
   $N_{G_1} := \text{ComputeOrdering}(G_1, G_2)$ 
   $s_0, \text{Parent} = \text{PreprocessPatternGraph}(G_1, N_{G_1})$ 
  Results := {}
  Match( $s_0, G_1, G_2, N_{G_1}, \text{Parent}, \text{Results}$ )
  return Results
end

```

**Fig. 1.** Outline of the VF3 algorithm. The VF3 function returns the set of solutions found.  $N_{G_1}$  is the *node exploration sequence* precomputed for  $G_1$ ,  $s_0$  is the initial state and **Parent** is a precomputed data structure used during the visit. The **Match** procedure is shown in Fig. 2.

A state is *consistent* if its partial mapping satisfies the constraints imposed by the required matching (subgraph isomorphism, in this case). A state represents a solution if it is consistent, and the mapping involves all the nodes in  $V_1$ . Since it can be demonstrated that a solution cannot be reached from an inconsistent state, the algorithm only generates consistent states in the search tree. For each state  $s$  the algorithm maintains the following information:

- $M(s) \subset V_1 \times V_2$ , the partial mapping; for the initial state  $s_0$ ,  $M(s_0) = \{\}$ ; we will denote as  $M_1(s)$  and  $M_2(s)$  the projections of  $M(s)$  onto  $V_1$  and  $V_2$  respectively;
- $\tilde{\mathcal{P}}_1(s) \subset V_1$  and  $\tilde{\mathcal{P}}_2(s) \subset V_2$ , the sets of nodes outside  $M(s)$  having an edge whose destination is a node in  $M_1(s)$  (for  $\tilde{\mathcal{P}}_1$ ) or in  $M_2(s)$  (for  $\tilde{\mathcal{P}}_2$ );

- $\tilde{\mathcal{S}}_1(s) \subset V_1$  and  $\tilde{\mathcal{S}}_2(s) \subset V_2$ , the sets of nodes outside  $M(s)$  having an edge whose origin is a node in  $M_1(s)$  (for  $\tilde{\mathcal{S}}_1$ ) or in  $M_2(s)$  (for  $\tilde{\mathcal{S}}_2$ ).

If the nodes have labels, VF3 can make use of them by partitioning the nodes into equivalence classes (each class corresponds to a disjoint subset of the labels) in order to speed up the search; in this case, the algorithm will keep for each state the projection of  $\tilde{\mathcal{P}}_1(s)$ ,  $\tilde{\mathcal{P}}_2(s)$ ,  $\tilde{\mathcal{S}}_1(s)$  and  $\tilde{\mathcal{S}}_2(s)$  onto each of the classes.

```

procedure Match( $s, G_1, G_2, N_{G_1}, \text{Parent}, \text{out Results}$ )
  if IsGoal( $s$ ) then append  $M(s)$  to Results
  else
    for  $(u_n, v_n) \in \text{NextCandidates}(s, N_{G_1}, \text{Parent}, G_1, G_2)$ 
      if IsFeasible( $s, u_n, v_n$ ) then
         $s_n := \text{ExtendState}(s, u_n, v_n)$ 
        Match( $s_n, G_1, G_2, N_{G_1}, \text{Parent}, \text{Results}$ )
        RestoreState( $s, u_n, v_n$ )
      end if
    end for
  end if
end

```

**Fig. 2.** The recursive match procedure. Here  $s$  is the search state,  $u_n$  and  $v_n$  are nodes evaluated for being added to the current partial mapping, and  $s_n$  is a new state obtained adding  $(u_n, v_n)$  to  $s$

An outline of the VF3 algorithm is given in Fig. 1. The algorithm, before commencing the depth-first visit of the search space, performs some preprocessing. First, the *node exploration sequence* for the nodes of the pattern graph ( $N_{G_1}$ , a permutation of  $V_1$ ) is defined, in order to explore first the nodes that are more rare and constrained, evaluating for each node  $u \in V_1$  the following criteria: the probability  $P_f(u)$  of finding a node  $v \in V_2$  that has the same label as  $u$  and a compatible degree (for subgraph isomorphism, the degree of  $v$  must be not smaller than that of  $u$ ); the number of connections of  $u$  to other nodes already inserted in the sequence  $N_{G_1}$ , since each connection becomes a constraint in the mapping; the degree of  $u$ , since nodes with larger degrees will introduce more constraints in the mapping.

After defining  $N_{G_1}$ , a preprocessing of  $G_1$  is performed to precompute, for each level of the search space, the following information:

- the sets  $\tilde{\mathcal{P}}_1(s)$  and  $\tilde{\mathcal{S}}_1(s)$ , since as shown in [9] they only depend on the depth level of  $s$ ;
- an associative array **Parent** that links each node of  $V_1$  the first node that is both connected to it and present in  $N_{G_1}$  before it;
- the initial state  $s_0$ , having an empty associated mapping.

After the preprocessing, the actual depth-first visit starts. Figure 2 shows the algorithm used for the visit, in the case that all the solutions are desired; the



algorithm is slightly different if only the first solution is requested. Each pair of nodes that is considered for addition to the current partial mapping, is examined using the **IsFeasible** function, described later, and if it passes this test, a new state  $s_n$  is built by extending  $s$ ; then the visit proceeds recursively on  $s_n$ . In order to save space, the data structures for  $s_n$  are not allocated from scratch; instead, the **ExtendState** function destructively reuses the data structures of  $s$ . Indeed, this allows VF3 to run with a space complexity that is linear in the number of nodes, as we will show in the next subsection. Because of this, after each recursive call, the **Match** procedure has to restore the previous condition of the data structures belonging to  $s$ ; this is done by the **RestoreState** procedure.

The **IsFeasible** function plays a central role in the algorithm: first, it checks if the addition of  $(u_n, v_n)$  will produce a new state that is consistent with the subgraph isomorphism constraints; furthermore, it includes the so-called *look-ahead* functions, that are heuristics to check if any consistent state can be reached in one or two steps from the obtained new state:

$$\text{IsFeasible}(s, u_n, v_n) = F_s(s, u_n, v_n) \wedge F_c(s, u_n, v_n) \wedge F_{la1}(s, u_n, v_n) \wedge F_{la2}(s, u_n, v_n) \quad (1)$$

where  $F_s$  is the *semantic feasibility function*, checking if  $u_n$  and  $v_n$  have the same labels and if the edges connecting them to  $M_1(s)$  and  $M_2(s)$  have the same labels.  $F_c$  checks the structural consistency of the new state: if an edge exists between  $u_n$  and a node in  $M_1(s)$ , an edge must also exist between  $v_n$  and the corresponding node in  $M_2(s)$ , and vice versa.  $F_{la1}$  is the 1-look-ahead function: it is a heuristic necessary condition that must be satisfied to ensure that at least one of the states derived by adding another pair of nodes to  $s_n$  is consistent; similarly  $F_{la2}$  is the 2-look-ahead function, regarding the states derived by adding two pairs of nodes to  $s_n$ . Notice that  $F_{la1}$  and  $F_{la2}$  are necessary but not sufficient conditions to ensure that a solution can be reached from  $s_n$ . For graphs without labels, the look-ahead functions are the following:

$$F_{la1}(s, u_n, v_n) \iff \begin{aligned} |\mathcal{P}_1(u_n) \cap \tilde{\mathcal{P}}_1(s)| &\leq |\mathcal{P}_2(v_n) \cap \tilde{\mathcal{P}}_2(s)| \\ |\mathcal{P}_1(u_n) \cap \tilde{\mathcal{S}}_1(s)| &\leq |\mathcal{P}_2(v_n) \cap \tilde{\mathcal{S}}_2(s)| \\ |\mathcal{S}_1(u_n) \cap \tilde{\mathcal{P}}_1(s)| &\leq |\mathcal{S}_2(v_n) \cap \tilde{\mathcal{P}}_2(s)| \\ |\mathcal{S}_1(u_n) \cap \tilde{\mathcal{S}}_1(s)| &\leq |\mathcal{S}_2(v_n) \cap \tilde{\mathcal{S}}_2(s)| \end{aligned} \quad (2)$$

$$F_{la2}(s, u_n, v_n) \iff \begin{aligned} |\mathcal{P}_1(u_n) \cap \tilde{V}_1(s)| &\leq |\mathcal{P}_1(v_n) \cap \tilde{V}_2(s)| \\ \wedge |\mathcal{S}_1(u_n) \cap \tilde{V}_1(s)| &\leq |\mathcal{S}_1(v_n) \cap \tilde{V}_2(s)| \end{aligned} \quad (3)$$

where  $\tilde{V}_1(s) = V_1 - M_1(s) - \tilde{\mathcal{S}}_1(s) - \tilde{\mathcal{P}}_1(s)$  and similarly  $\tilde{V}_2(s) = V_2 - M_2(s) - \tilde{\mathcal{S}}_2(s) - \tilde{\mathcal{P}}_2(s)$ . In the case of labeled graphs the sets  $\tilde{\mathcal{S}}_i(s)$  and  $\tilde{\mathcal{P}}_i(s)$  are kept separately for each equivalence class into which the node labels are divided, and so the above equations are replicated for each class.

## 2.2 VF3-Light: Removing the Look-Ahead Rules

The look-ahead functions described by Eqs. 2 and 3 are not needed to ensure the correctness of the found solutions. Without them, the algorithm would find exactly the same solutions, but will possibly have to explore more states to reach them. The same is true for the reordering of the nodes of the pattern graph: the algorithm would be correct with whatever order of the nodes, but the one chosen in VF3 aims at introducing as soon as possible the nodes that have more constraints, so as to discard earlier unfruitful portions of the state space. The combined effects of these two heuristics results in the high performance shown by VF3 on large and dense graphs [8]. However, we decided to investigate if on simple graphs these two heuristics may be somewhat redundant.

The node reordering does not require the use of additional data structures, and does not take time during the recursive visit of state space. Conversely, for computing the look-ahead functions the algorithm needs to keep the  $\tilde{\mathcal{P}}_2(s)$  and  $\tilde{\mathcal{S}}_2(s)$  sets for each state  $s$  (as we said earlier,  $\tilde{\mathcal{S}}_1(s)$  and  $\tilde{\mathcal{P}}_1(s)$  can be precomputed). In principle, these sets could occupy a memory that is  $O(N_2)$  (where  $N_1$  and  $N_2$  are the number of nodes in  $G_1$  and  $G_2$ ). Since the depth-first visit of the tree keeps in memory at most  $O(N_1)$  states, the memory requirement would be  $O(N_1 \cdot N_2)$ . However, in the implementation of VF3 we have reused the data structure of the parent state when a child state is derived from it, restoring its original content when the exploration of the child is finished. Thus, the overall memory occupation remains  $O(N_2)$ .

On the other hand, the time needed to compute  $\tilde{\mathcal{S}}(s_n)$  and  $\tilde{\mathcal{P}}(s_n)$  from the corresponding sets of  $s$  is proportional to the degrees of  $u_n$  and  $v_n$ , and must be spent for each new state that is visited. A similar time is needed to restore the previous content of the data structures when the visit of the state is finished. So, in the trade-off between the number of visited states and the time spent on each state, it is entirely possible that the use of the feasibility rules may worsen the performance of the algorithm on those graphs where the reordering heuristic already removes most of the unfruitful paths. To verify that this is the case, we

**Table 1.** Characteristics of the datasets used to benchmark VF3-light

Dataset	Graphs	Target size	Pattern size	Labels
MIVIA BVG	6000	20–1000 nodes	20% of target size	-
MIVIA M2D	4000	16–1024 nodes	20% of target size	-
MIVIA M3D	3200	27–1000 nodes	20% of target size	-
MIVIA M4D	2000	16–1096 nodes	20% of target size	-
MIVIA RAND	3000	20–1000 nodes	20% of target size	-
Proteins	300	535–10081 nodes	8–256	4–5
Molecules	10000	8–99 nodes	8–64	4–5
Scale-free	100	200–1000 nodes	90% of target size	-

have defined and implemented a modified algorithm, called *VF3-Light*, which has the following modifications with respect to VF3:

- removal of the computation of  $\tilde{S}_1$  and  $\tilde{P}_1$  in the preprocessing phase;
- removal of  $\tilde{S}_2(s)$  and  $\tilde{P}_2(s)$  from the state data structure, and of their computation and restoring in `ExtendState` and `RestoreState`;
- removal of  $F_{la1}$  and  $F_{la2}$  from `IsFeasible`.

### 3 Experiments

Due to the complexity and variety of subgraph graph isomorphism there is no single algorithm that is able to outperform the others for all the possible kind of graphs and applications. For this reason, we have chosen a group of datasets that, at the same time, contain different graph families and are representative to some relevant fields applications of subgraph isomorphism, i.e. biology and social networks. The first dataset is the MIVIA [5, 10], which is well-known and widely used; it is composed of more that 10000 unlabeled graphs belonging to three main typologies: bounded valence, random graphs and open meshes (regular and irregular). This dataset was proposed more than ten years ago to profile the performance of VF2, but is still considered an important benchmark for any new exact graph matching method [11]. Additionally, we have considered two biological datasets of graphs extracted from real protein and molecule structures, proposed during the International Contest on Graph Matching Algorithms for Pattern Search in Biological Databases hosted by the ICPR 2014 [12]; and a synthetic dataset of scale-free graphs, proposed by Solnon in [13, 14], generated using the Barabási-Albert model [15], that is representative both of social networks and of protein-protein interaction networks. In Table 1 we briefly show the characteristics of these datasets. The experiments have been conducted on a cluster infrastructure with VMWare ESXi 5. All the virtual machines have been configured with two dedicated AMD Opteron running at 2,300 MHz, with 2 Mb of cache and 4 Gb of RAM.

**Table 2.** Overall execution time of the algorithms on each dataset. *Time* is the matching time in seconds; *relative time* is the ratio between the time of the algorithm and the one of the fastest algorithm on the same dataset.

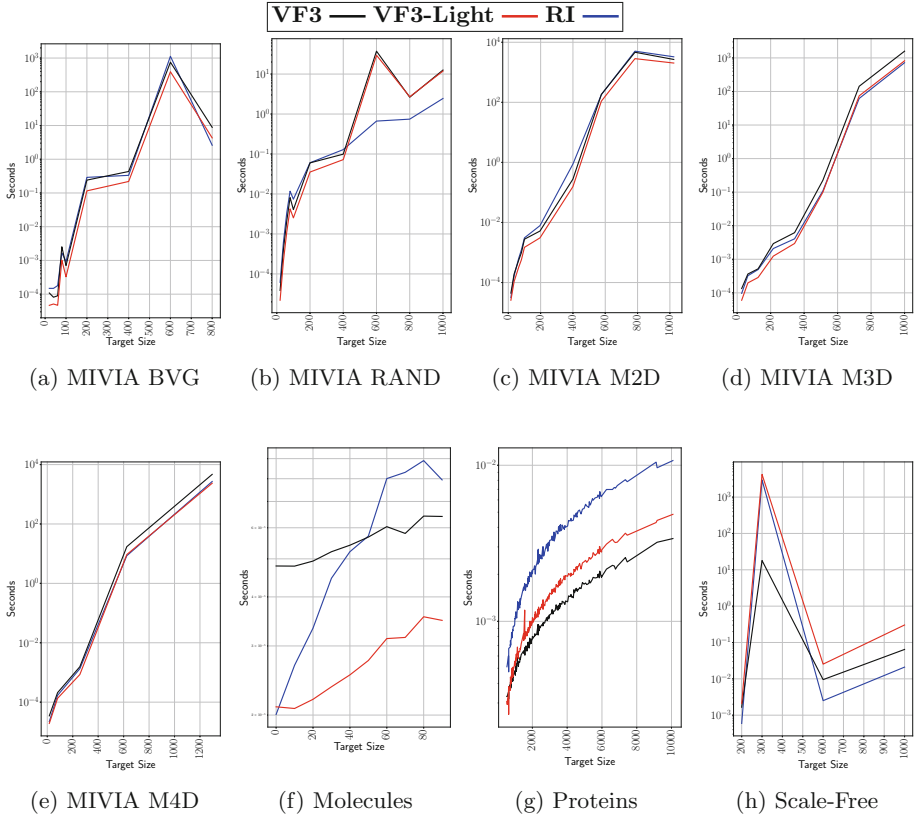
	VF3		VF3-Light		RI	
	Time	Relative Time	Time	Relative Time	Time	Relative Time
BVG	1.41e+05	1.92	7.33e+04	1.00	2.10e+05	2.87
RAND	1.58e+04	12.96	1.33e+04	10.87	1.22e+03	1.00
M2D	9.02e+05	1.63	5.55e+05	1.00	9.76e+05	1.76
M3D	6.89e+05	2.22	3.56e+05	1.15	3.11e+05	1.00
M4D	1.33e+05	1.98	6.73e+04	1.00	7.62e+04	1.13
Molecules	2.25e+01	2.19	1.02e+01	1.0	2.30e+01	2.24
Proteins	1.94e+01	1.0	2.62e+01	1.35	5.69e+01	2.93
Scale-Free	6.32e+02	1.00	1.48e+05	233.65	1.04e+05	164.09

**Table 3.** Matching time vs target size on the MIVIA datasets. For each kind of graphs, *time* is the average matching time in seconds; *relative time* is the ratio between the average matching time of the algorithm and that of the fastest algorithm for the same target size.

	Size	VF3		VF3-Light		RI	
		Time	Relative Time	Time	Relative Time	Time	Relative Time
BVG	80	2.54e-03	2.49	1.02e-03	1.00	1.67e-03	1.64
	100	7.06e-04	2.16	3.26e-04	1.00	9.32e-04	2.86
	200	2.41e-01	2.08	1.15e-01	1.00	2.90e-01	2.52
	400	4.34e-01	1.98	2.19e-01	1.00	3.33e-01	1.52
	600	7.54e+02	1.92	3.93e+02	1.00	1.13e+03	2.87
	800	8.82e+00	3.39	4.30e+00	1.65	2.60e+00	1.00
RAND	80	8.13e-03	1.91	4.25e-03	1.00	1.18e-02	2.77
	100	4.07e-03	1.61	2.52e-03	1.00	7.40e-03	2.93
	200	6.00e-02	1.69	3.54e-02	1.00	6.04e-02	1.71
	400	9.91e-02	1.37	7.23e-02	1.00	1.29e-01	1.78
	600	3.74e+01	56.12	2.96e+01	44.39	6.66e-01	1.00
	800	2.63e+00	3.53	2.71e+00	3.63	7.45e-01	1.00
	1000	1.26e+01	5.15	1.19e+01	4.85	2.45e+00	1.00
M2D	81	9.81e-04	1.72	5.70e-04	1.00	1.22e-03	2.14
	100	2.77e-03	1.87	1.49e-03	1.00	3.08e-03	2.07
	196	5.18e-03	1.69	3.07e-03	1.00	7.84e-03	2.55
	400	2.78e-01	1.78	1.56e-01	1.00	8.84e-01	5.67
	576	1.83e+02	1.67	1.10e+02	1.00	1.81e+02	1.65
	784	4.64e+03	1.63	2.85e+03	1.00	5.05e+03	1.77
	1024	2.68e+03	1.32	2.03e+03	1.00	3.28e+03	1.61
M3D	64	3.64e-04	1.84	1.98e-04	1.00	3.24e-04	1.64
	125	5.19e-04	1.81	2.87e-04	1.00	4.93e-04	1.72
	216	2.93e-03	2.36	1.24e-03	1.00	2.09e-03	1.68
	343	6.21e-03	2.10	2.96e-03	1.00	4.07e-03	1.38
	512	2.25e-01	2.26	9.95e-02	1.00	1.09e-01	1.09
	729	1.43e+02	2.31	7.42e+01	1.20	6.20e+01	1.00
	1000	1.59e+03	2.21	8.20e+02	1.14	7.19e+02	1.00
M4D	16	3.46e-05	1.80	1.92e-05	1.00	2.22e-05	1.16
	81	2.09e-04	1.55	1.35e-04	1.00	1.69e-04	1.26
	256	1.56e-03	1.83	8.51e-04	1.00	1.33e-03	1.57
	625	1.72e+01	2.02	9.34e+00	1.09	8.53e+00	1.00
	1296	4.68e+03	1.99	2.36e+03	1.00	2.70e+03	1.15

We have compared VF3-Light against VF3 [9] and RI [11], a three-search based algorithm approaching subgraph isomorphism without look-ahead, similarly to our algorithm, but with different heuristics and sorting procedure. The matching times for the three considered algorithms to find all the subgraph isomorphism solutions are shown in Figs. 3a–h. Table 2 show the overall matching time for each algorithm on each entire dataset. Table 3 provides more detailed information on the matching times with respect to target size. In these tables, beside the absolute value of the matching times, we have also reported the *relative times*, normalized with respect to the fastest time (e.g. 1 means the fastest time, 1.3 means 30% longer than the fastest time and so on).

As we expected, VF3, which is designed to deal very large and dense graphs (more than a thousand nodes), is confirmed to be the most effective algorithm on large labelled graphs extracted from protein (Fig. 3g), where it outperforms both VF3-Light and RI (that are respectively 35% and almost 200% slower).



**Fig. 3.** The total matching times on each dataset.

Similarly, on scale-free graphs (Fig. 3h), that are dense random graphs generated using a power law distribution of degrees [15], the full VF3 is again considerably faster than VF3-Light and RI, by more than two orders of magnitude. On this dataset, for some of the graphs RI turns out to outperform both, but on the hardest graphs VF3 is by a large margin the fastest algorithm, thus yielding a much shorter overall matching time. On the remaining datasets, VF3-Light is always faster than the full VF3. In particular, it becomes significantly faster on Bounded Valence graphs (Fig. 3a), 2D/3D/4D meshes (Fig. 3c, d and e) and molecules (Fig. 3f), where VF3 requires a time that is respectively 92%, 63%, 93%, 98% and 112% longer than VF3-Light. Moreover, on Bounded Valence graphs, 2D meshes and molecules, VF3-Light is also able to significantly outperform RI (being 187%, 76% and 124% faster), resulting the fastest algorithm. On the other hand, on the MIVIA Random graphs RI is faster than VF3-Light by an order of magnitude, and on 3-D and 4-D meshes these two algorithms are quite close to each other (about 15% of difference).

From the exam of Table 3, we can see that VF3-Light always result the fastest algorithm of the three for small to medium-sized graphs (up to about 500 nodes). Notice that on Random graphs there is an anomaly at 600 nodes: a single pattern/target pair that makes the average the matching time of both VF3 and VF3-Light considerably longer. We will have to better study this particular pair, understanding why it is so problematic for our algorithms, in order to further improve their heuristics.

## 4 Conclusions

In this paper we have introduced VF3-Light, a subgraph isomorphism algorithm obtained by removing some of the heuristics used in VF3, namely the so called look-ahead functions. The removal of these heuristics makes the algorithm faster in the visit of each search state, but also implies that a larger number of states may need to be visited for finding the solutions. An experimental evaluation on several kinds of graphs shows that indeed on very large or very dense graphs, for which the VF3 algorithm was designed, the look-ahead heuristics give an advantage, but on other, simpler kinds of graphs VF3-Light is able to outperform VF3. These are only the first results obtained on the new algorithm; further experiments will be performed in the future in order to provide a more precise characterization of the situations where the balance is in favor of either VF3 or VF3-Light, so as to give the users some criteria for deciding which algorithm to choose for a given application problem.

## References

1. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. *Int. J. Pattern Recogn. Artif. Intell.* **18**(3), 265–298 (2004)
2. Foggia, P., Percannella, G., Vento, M.: Graph matching and learning in pattern recognition on the last ten years. *Int. J. Pattern Recogn. Artif. Intell.* **28**(1), 1450001 (2014)
3. Vento, M.: A long trip in the charming world of graphs for pattern recognition. *Pattern Recogn.* **48**, 1–11 (2014)
4. Ullmann, J.R.: An algorithm for subgraph isomorphism. *J. Assoc. Comput. Mach.* **23**, 31–42 (1976)
5. Cordella, L., Foggia, P., Sansone, C., Vento, M.: A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**, 1367–1372 (2004)
6. Almasri, I., Gao, X., Fedoroff, N.: Quick mining of isomorphic exact large patterns from large graphs. In: *IEEE International Conference on Data Mining Workshop*, pp. 517–524, December 2014
7. Bonnici, V., Giugno, R.: On the variable ordering in subgraph isomorphism algorithms. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **14**(1), 193–203 (2017)
8. Carletti, V., Foggia, P., Saggese, A., Vento, M.: Challenging the time complexity of exact subgraph isomorphism for huge and dense graphs with VF3. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(4), 804–818 (2018)

9. Carletti, V., Foggia, P., Saggese, A., Vento, M.: Introducing VF3: a new algorithm for subgraph isomorphism. In: Foggia, P., Liu, C.L., Vento, M. (eds.) GBRPR 2017, pp. 128–139. Springer International Publishing, Cham (2017). <https://doi.org/10.1007/978-3-319-58961-9-12>
10. MIVIA Lab: MIVIA dataset and MIVIA large dense graphs dataset (2017). <http://mivia.unisa.it/>
11. Bonnici, V., Giugno, R., Pulvirenti, A., Shasha, D., Ferro, A.: A subgraph isomorphism algorithm and its application to biochemical data. *BMC Bioinform.* **14**, S13 (2013)
12. Carletti, V., Foggia, P., Vento, M., Jiang, X.: Report on the first contest on graph matching algorithms for pattern search in biological databases. In: GBR 2015, pp. 178–187 (2015)
13. Kotthoff, L., McCreesh, C., Solnon, C.: Portfolios of subgraph isomorphism algorithms. In: Festa, P., Sellmann, M., Vanschoren, J. (eds.) LION 2016. LNCS, vol. 10079, pp. 107–122. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-50349-3\\_8](https://doi.org/10.1007/978-3-319-50349-3_8)
14. Solnon, C.: Solnon datasets (2017). <http://liris.cnrs.fr/csolnon/SIP.html>
15. Barabási, A.-L., Oltvai, Z.N.: Network biology: understanding the cell's functional organization. *Nat. Rev. Genet.* **5**(2), 101–113 (2004)



# A Deep Neural Network Architecture to Estimate Node Assignment Costs for the Graph Edit Distance

Xavier Cortés<sup>1</sup>(✉), Donatello Conte<sup>1</sup>, Hubert Cardot<sup>1</sup>,  
and Francesc Serratosa<sup>2</sup>

<sup>1</sup> LiFAT, Université de Tours, Tours, France  
{xavier.cortes, donatello.conte,  
hubert.cardot}@univ-tours.fr

<sup>2</sup> Universitat Rovira i Virgili, Tarragona, Catalonia, Spain  
francesc.serratosa@urv.cat

**Abstract.** The problem of finding a distance and a correspondence between a pair of graphs is commonly referred to as the Error-tolerant Graph matching problem. The Graph Edit Distance is one of the most popular approaches to solve this problem. This method needs to define a set of parameters and the cost functions aprioristically. On the other hand, in recent years, Deep Neural Networks have shown very good performance in a wide variety of domains due to their robustness and ability to solve non-linear problems. The aim of this paper is to present a model to compute the assignments costs for the Graph Edit Distance by means of a Deep Neural Network previously trained with a set of pairs of graphs properly matched. We empirically show a major improvement using our method with respect to the state-of-the-art results.

## 1 Introduction

Graphs are defined by a set of nodes (local components) and edges (the structural relations between them), allowing to represent the connections that exist between the component parts of an object. Due to this, graphs have become very important to model objects that require this kind of representation. In fields like cheminformatics, bioinformatics, computer vision and many others, graphs are commonly used to represent objects [1].

One of the key points in pattern recognition is to define an adequate metric to estimate distances between two patterns. The Error-tolerant Graph Matching tries to address this problem. In particular, the Graph Edit Distance (GED) [2] is an approach to solve the Error-tolerant Graph Matching problem by means of a set of edit operations including insertions, deletions and node assignments, also referred to as node substitutions. On the other hand, Deep Neural Networks (DNNs) have become a very powerful tool applied in several domains due to their ability to find models.

The aim of this paper is to propose a new way to estimate node assignment costs for GED, using a DNN trained with a set of graphs correspondences properly labelled. The document is organized as follows: in Sect. 2 are presented the definitions to understand



the paper, in Sect. 3 is presented the state-of-the-art, in Sect. 4 we describe the architecture and de details of our model while Sect. 5 shows the experimental results. Finally, the conclusions are presented in Sect. 6.

## 2 Definitions and Methods

### 2.1 Attributed Graph

Formally, we define an attributed graph as a quadruplet  $G = (\Sigma_v, \Sigma_e, \gamma_v, \gamma_e)$ , where  $\Sigma_v = \{v_i | i = 1, \dots, n\}$  is the set of nodes,  $\Sigma_e = \{e_{ij} | i, j \in 1, \dots, n\}$  is the set of edges connecting pairs of nodes,  $\gamma_v$  is a function to map nodes to their attributed values and  $\gamma_e$  maps the structure of the nodes.

### 2.2 Graphs Correspondence

We define a correspondence between two graphs  $G^p$  and  $G^q$  as a set of assignments  $f : \Sigma_v^p \rightarrow \Sigma_v^q$  that univocally relate the nodes of  $G^p$  to the nodes of  $G^q$ . Where  $f(v_i^p) = v_j^q$  if exist the assignment  $v_i^p \rightarrow v_j^q$ .

### 2.3 Node Assignment Costs for the Graphs Edit Distance

The basic idea of the GED [2] between two graphs  $G^p$  and  $G^q$ , is to find the minimum cost to transform completely  $G^p$  into  $G^q$  by means of a set of edit operations, including insertions, deletions and node assignments, commonly referred to as *editpath*. Cost functions are introduced to quantitatively evaluate the level of distortion that each edit operation introduces.

$$c(v_i^p \rightarrow v_j^q) = c_v(v_i^p \rightarrow v_j^q) + c_e(v_i^p \rightarrow v_j^q) \tag{1}$$

The cost of an assignment edit operation (1) is typically given by the distance measure between the nodes attributes  $c_v(v_i^p \rightarrow v_j^q) = \text{local\_distance}(\gamma_v^p(v_i^p), \gamma_v^q(v_j^q))$  and by the cost of substituting the local structures  $c_e(v_i^p \rightarrow v_j^q) = \text{structural\_distance}(\gamma_e^p(v_i^p), \gamma_e^q(v_j^q))$ . These cost functions estimate the degree of separation between a pair of nodes  $v_i^p$  and  $v_j^q$  belonging to graphs  $G^p$  and  $G^q$ . The Euclidean distance is a common way to estimate the *local\_distance* between the nodes attributes, while in [3] are presented different metrics to estimate the *structural\_distance*. Our model, as we will see, automatically learns the costs of these assignments from a set of training correspondences previously labeled without having to define the cost functions.

In order to allow the maximum flexibility in the matching process and taking into account that graphs can have different cardinality and that a node that appears in  $G^p$  could not be in  $G^q$ , graphs can be extended with null nodes adding penalty costs when

an existing node of one graph is assigned to a null one of the other graph. In this paper we do not consider this option since we focus on the problem of node assignments comparing our results with other works that face the same problem, as in [4, 5]. However, our model can be easily combined with other models that consider null nodes by adding penalty costs for insertions and deletions.

## 2.4 Hamming Distance

The hamming distance is a metric to compare graph correspondences used typically to assess the correctness of a correspondence comparing the correspondence that we are evaluating with respect to the ground-truth one. This metric evaluates the ratio between the number of correct assignments and the total number of assignments in the evaluated correspondence. Formally:

Let  $f : \Sigma_v^p \rightarrow \Sigma_v^q$  the automatic correspondence and  $f' : \Sigma_v^{p'} \rightarrow \Sigma_v^{q'}$  the ground-truth correspondence between two graphs  $G^p$  and  $G^q$  with cardinality  $n$  (graphs can be extended with null nodes to manage insertions or deletions of nodes), the hamming distance is formally defined as:

$$\Delta^h(f, f') = \frac{\sum_{i=1}^n (1 - \delta(f(v_i^p), f'(v_i^{p'})))}{n} \quad (2)$$

Where,  $\delta$  is the Kronecker Delta function:

$$\delta(a, b) = \begin{cases} 0, & \text{if } a \neq b \\ 1, & \text{if } a = b \end{cases} \quad (3)$$

## 2.5 Deep Neural Networks

DNNs are a computational model inspired by the neural networks existing in many biological organisms [6]. They have become very popular in many fields due to its adaptability and learning capacity.

The classical architecture of a DNN consists of an input layer, an output layer and a cascade of multiple hidden layers in the middle. Each layer contains several neurons connected with the neurons of the previous layer. The connections between neurons have different weights fixing the strength of the signal at the connection. Each neuron executes an activation function having as inputs the values of the connections with the previous layer and sending the output to the neurons of the next layer. The signal path goes from the input layer to the output layer. Depending on the connections weights and the bias values, the output can be different given the same input.

During the training process the learning algorithm adjust the weights and bias according to the values of a training set trying to minimize the error between the given inputs and the expected outputs.

### 3 State of the Art

The distance value of the GED depends on the edit costs, in particular  $c_v$  (distance between the nodes attributes),  $c_e$  (distance between the local structures) and the penalties costs for insertions and deletions. Typically, these costs must be defined and parameterized aprioristically. Depending on how these parameters and costs functions are defined the performance in terms of hamming distance between the automatically deduced correspondence and a ground truth correspondence or graphs classification accuracy, can be different.

Recently, in order to maximize the performance of different Error-Tolerant Graph Matching approaches, some researchers have focused their work on automatically learn the parameters and the cost functions instead of using the traditional trial-error method.

We can divide the learning methods in three main groups depending on the objective function. The first group [7–10] addresses the recognition ratio for graph classification, while the second group [4, 5, 11, 12] targets the hamming distance. Finally, there is a special case in [13] that does not learn the parameters to estimate the costs but tries to predict if an assignment between nodes is correct or not depending on the values of the costs matrix (the matrix with the costs of each edit operation). Moreover, another subdivision can be considered depending if the methods try to learn the assignments costs or the insertions and deletions. The aim of our paper is to propose a model to estimate only the assignments costs minimizing the hamming distance, as in [4, 5]. As we have commented before, our model can be combined with other models that consider nodes insertions and deletions but we do not address this particularity in this paper.

### 4 Proposed Architecture

In this section we describe a new architecture based on DNNs to estimate assignments costs (Sect. 2.3) between a pair of nodes by means of a DNN (Sect. 2.5) in order to minimize the hamming distance (Sect. 2.4).

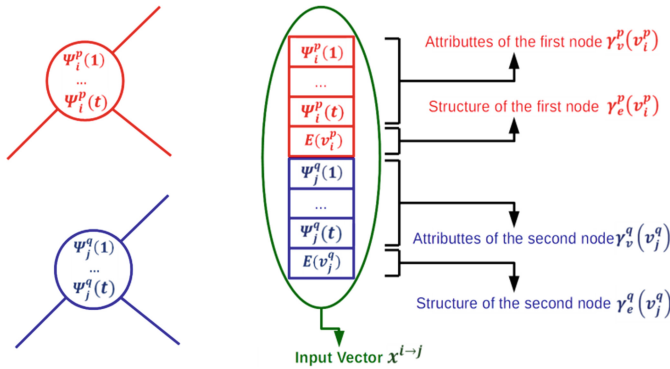
$$c(v_i^p \rightarrow v_j^q) = \text{DNN}(v_i^p \rightarrow v_j^q) \tag{4}$$

#### 4.1 Node Assignment Embedding

The first step of our model consists of transforming the local and structural information of both nodes into a set of inputs for the network. In this section we show how to embed this information into an input vector.

Let  $G^p$  and  $G^q$  two attributed graphs,  $\gamma_v^p = \{v_i^p \rightarrow \Psi_i^p | i = 1..n\}$  a function that assigns  $t$  attribute values from an arbitrary domain to each node of  $G^p$ , where  $\Psi_i^p \in \mathbb{R}^t$  is defined in a metric space of  $t \in \mathbb{R}$  dimensions and  $\gamma_e^p = \{v_i^p \rightarrow E(v_i^p) | i = 1..n\}$  where  $E(\cdot)$  refers to the number of edges of a certain node (the Degree centrality [3]). And similar for  $\gamma_v^q$  and  $\gamma_e^q$  in  $G^q$ .

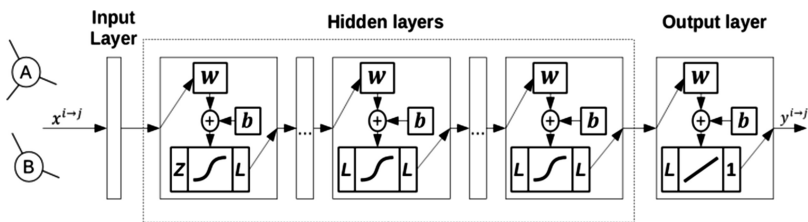
Vector  $x^{i \rightarrow j} = [\gamma_v^p(v_i^p), \gamma_e^p(v_i^p), \gamma_v^q(v_j^q), \gamma_e^q(v_j^q)] \in \mathbb{R}^{(t+1) \cdot 2}$  is the embedded representation of the assignment  $v_i^p \rightarrow v_j^q$  where each position of the vector  $x^{i \rightarrow j}$  corresponds to one of the values of the input layer of the DNN that estimates the assignment cost between the node  $v_i^p$  of  $G^p$  and the node  $v_j^q$  of  $G^q$  (Fig. 1).



**Fig. 1.** An illustration showing the embedding process of two nodes (red and blue) into an input vector. (Color figure online)

### 4.2 Network Architecture

The topology we propose is a classical topology for parameters fitting consisting of a multi-layer network using the sigmoid activation function for the hidden layers and a linear function for the output layer (Fig. 2). In the experimental section we show the results achieved with different configurations changing the number of neurons and the number hidden layers.

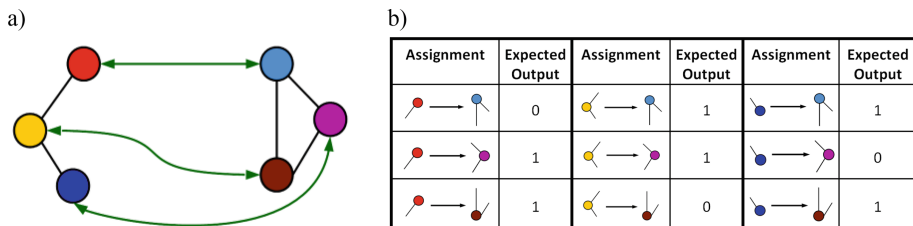


**Fig. 2.** DNN architecture for node assignments costs.  $Z$  is the number of inputs (size of the vector  $x^{i \rightarrow j}$ ).  $L$  the number of neurons of each hidden layer,  $w$  the weights and  $b$  the bias.

The input of the network representing the nodes to be assigned is the vector  $x^{i \rightarrow j} \in \mathbb{R}^{(t+1) \cdot 2}$  (defined in Sect. 4.1) and the output is a real value theoretically defined within a cost range from zero to one viz.  $y^{i \rightarrow j} = \{c \in \mathbb{R} : 0 \leq c \leq 1\}$ . Zero is the expected value when there is no penalty for the assignment and one is the maximum expected value penalizing a node assignment.

### 4.3 Training the Model

We manage the problem of training the DNN as a supervised learning problem. The training set has  $K$  observations. Each observation is composed of a triplet consisting of pair of graphs and the correspondence that relates its nodes  $\{G^{p^k}, G^{q^k}, f^k\}$ . The ground-truth correspondences  $f^k$  must be provided by an oracle according to the problem (images, fingerprints, letters...).



**Fig. 3.** (a) Correspondence between a pair of graphs. Colored circles: Nodes. Black lines: Edges. Green arrows: Graphs correspondence. (b) Set of all possible node assignments and expected DNN outputs given the correspondence in (a). (Color figure online)

Then, assuming that the assignment cost must be low if two nodes are matched and high in the opposite case and taking into account that the outputs range goes from zero to one (Sect. 4.2), we propose to feed the learning algorithm with a set of  $R$  inputs-outputs pairs  $\{x^{v_i^{p^k} \rightarrow v_j^{q^k}}, o^r\}$  that we deduce from the training set  $\{G^{p^k}, G^{q^k}, f^k\}$ . Where  $v_i^{p^k}$  and  $v_j^{q^k}$  are two nodes belonging to graphs  $G^{p^k}$  and  $G^{q^k}$  respectively.  $x^{v_i^{p^k} \rightarrow v_j^{q^k}}$  are the inputs of the DNN representing the assignment between  $v_i^{p^k}$  and  $v_j^{q^k}$  (Sect. 4.1). And  $o^r$  is the expected output, zero if  $f^k(v_i^{p^k}) = v_j^{q^k}$  and one otherwise.

In Fig. 3b, we show the expected outputs between nodes when the ideal correspondence is the correspondence shown in Fig. 3a. Zero when there is an assignment in the ground-truth correspondence and one when not. Note that there are more cases in which the expected output must be one because the correspondences between graphs are bijective by definition in our framework. That means, each node of  $G^{p^k}$  is assigned to a single node of  $G^{q^k}$  while it is unassigned to all the other nodes. For this reason and in order to prevent unbalancing problems we propose to oversample the positive assignments between nodes (when the expected output is zero) repeating them in the set of inputs-outputs that feeds the learning algorithm  $n - 1$  times, where  $n$  is the graphs cardinality.

The training algorithm used to learn the bias and weights of the network is the Leveberg-Marquardt [14].

#### 4.4 Graph Matching Algorithm

The graph matching method we propose is inspired by the Bipartite-GED [15] which is one of the most popular methods used to reduce the computational complexity of the GED problem to a Linear Sum Assignment Problem (LSAP). First, we build a cost matrix in which each cell corresponds to the cost of an assignment. The algorithm fills the values of this matrix with the DNN outputs. Our algorithm does not extend the matrix for insertions and deletions since we only consider the assignments between nodes. The process of assigning nodes can be solved as a LSAP on C matrix. In our experiments we used the Hungarian [16] solver. The final step is to sum the costs of the solution provided by the solver.

---

**Algorithm:** Neural Graph Matching

---

**Input:** Graph G1, G2; DNN network;  
**Output:** Correspondences Co; Cost Ct;

```

1: Initialisation:
2: foreach Node NodeI of G1
3:   foreach Node NodeJ of G2
4:     x:=inputVector(NodeI,NodeJ);
5:     y:=computeCosts(network,x);
6:     C(I,J) = y;
7:   end
8: end
9: [Co, Ct] = solveLSAP(C);

```

---

**Algorithm 1.** Learning Graph Matching methods.

## 5 Experiments

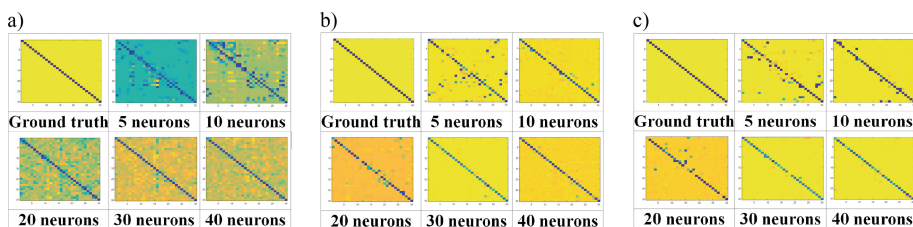
We divided the experimental section in three parts. First, we describe the database used in the experiments. Second, we show the resultant costs matrix using different network configurations. Finally, we present the hamming distance results using our model compared with the state-of-the-art algorithms that face the same kind of problem.

### 5.1 Databases

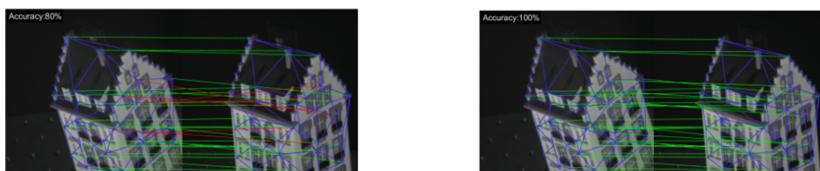
The HOUSE-HOTEL database described in detail in [17] consists of two sequences of frames showing two computer modeled objects, 111 frames of a HOUSE and 101 frames of a HOTEL, rotating on its own axis. Each frame of these sequences has the same 30 salient points identified and labelled. Each salient point represents a node of the graph and it is attributed by 60 Context Shape features. They triangulated the set of salient points using the Delaunay triangulation to generate the structure of the graphs. They made three sets of frames pairs taking into account different baselines (number of frames of separation in the video sequence). One set was used to learn, another to validate and the third one to test the model. Since the salient points are labelled we know the ground-truth correspondence between the nodes of the graphs.

## 5.2 Costs Matrix

This section shows the heatmaps of the resultant costs matrix (C matrix in Sect. 4.4) using our model. The aim of this experiment is to find a cost matrix minimizing the costs when the nodes must be assigned and maximizing the costs when not. Since we know the ground-truth correspondence we can deduce the ground-truth cost matrix. Figure 4a shows the results using a single hidden layer while Fig. 4b shows the same results using 5 hidden layers and Fig. 4c shows the results using 10 hidden layers with different configurations of numbers of neurons per layer. Blue color represents low costs values while yellow color represents high costs values. The experiment was performed using the first pair of graphs of the test set in the HOUSE sequence separated by 90 frames and the model has been trained with all the graphs separated by 90 frames in the training set.



**Fig. 4.** Costs matrix heatmaps between two graphs corresponding to the HOUSE dataset (90 frames of separation) using (a) 1 hidden layer, (b) 5 hidden layers and (c) 10 hidden layers. (Color figure online)



**Fig. 5.** Correspondences found between two graphs of the HOTEL sequence using our model. Left: single-layer and 10 neurons per layer, Right: five-layers and 10 neurons per layer. Blue lines are the edges between these nodes. Green lines: correct assignments. Red lines: incorrect assignments. (Color figure online)

We observe how the model tends to separate better the correct assignments from the incorrect ones when we increase the number of neurons and layers until reaching a point where the improvement is no longer increasing and even it could decrease. This can be explained because when we increase the network complexity, the model is able to find deeper non-linear correlations between the attributes that feature the nodes, but reached a critical point, could present overfitting problems due to there are more neurons than the ones that can be justified by the data.

Figure 5 shows the obtained correspondences computing a cost matrix with a single-layer (*left*) and with five-layers (*right*) of 10 neurons each layer in order to illustrate the performance of the model with different network configurations in terms of matching accuracy.

### 5.3 Hamming Distance Results

The main goal of our model is to reduce the hamming distance performing the GED. In the following experiment we show the hamming distance results between the correspondence found by our model and the ground-truth correspondence. In Table 1, we compare our results with respect to the state-of-the-art, note that smaller values mean better performance. We train, validate and test the model using different pairs of graphs as we described in Sect. 5.1. The baseline of our experiments is the number of frames of separation in the video sequence. Since the objects are in motion, consecutive frames are more similar than the distant ones. Therefore, the problem tends to be more complex when we increase the number of frames of separation. A single-layer network with 30 neurons per layer has been enough to reduce the hamming distance to zero for all the experiments, however, in Fig. 4, we show how deeper networks tend to increase the gap between the costs, generally separating better the correct assignments from the incorrect ones. The achieved results using our model represent a major improvement with respect to the previously presented results. We discuss the results in the next section.

**Table 1.** Hamming distance results on House and Hotel datasets.

House				Hotel			
#Frames	[4]	[5]	Our model	#Frames	[4]	[5]	Our model
90	0.14	0.24	<u>0</u>	90	0.09	0.21	<u>0</u>
80	0.14	0.18	<u>0</u>	80	0.17	0.18	<u>0</u>
70	0.13	0.10	<u>0</u>	70	0.14	0.15	<u>0</u>
60	0.09	0.06	<u>0</u>	60	0.13	0.16	<u>0</u>
50	0.19	0.04	<u>0</u>	50	0.09	0.07	<u>0</u>
40	0.02	0.02	<u>0</u>	40	0.07	0.04	<u>0</u>
30	0.02	0.01	<u>0</u>	30	0.04	0.02	<u>0</u>
20	0.01	<u>0</u>	<u>0</u>	20	0.02	<u>0</u>	<u>0</u>
10	<u>0</u>	<u>0</u>	<u>0</u>	10	<u>0</u>	<u>0</u>	<u>0</u>

*\*Results obtained with 1 layer of 30 neurons*

## 6 Conclusions

We have presented a new model to estimate assignment costs for the Graphs Edit Distance using a Deep Neural Network. We experimentally show that our model is able to find the ideal solution independently of the number of frames of separation. These



results represent a major improvement with respect to the previous state-of-the-art results, in particular, when the number of frames of separation is large. This means that the model can manage important distortions in the representations when it tries to find the best correspondence. We conclude that the improvement is because using neural networks allows to find multiple correlations between nodes attributes when performing the matching and our model is not limited by having to define a particular distance metric aprioristically since it learns the costs functions.

We consider that this work represents an important step to define the costs functions for node assignments in the problem of the Graph Edit Distance. However it is necessary to train the network with a set of examples properly labeled. The next step is to expand the model including insertions and deletions costs.

**Acknowledgments.** This work is part of the LUMINEUX project supported by the Region Centre-Val de Loire (France) and by the Spanish projects TIN2016-77836-C2-1-R and ColRobTransp MINECO DPI2016-78957-R AEI/FEDER EU; and also, the European project AEROARMS, H2020-ICT-2014-1-644271.

## References

1. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. *Int. J. Pattern Recogn. Artif. Intell.* **18**(3), 265–298 (2004)
2. Bunke, H., Allermann, G.: Inexact graph matching for structural pattern recognition. *Pattern Recogn. Lett.* **1**(4), 245–253 (1983)
3. Serratos, F., Cortés, X.: Graph edit distance: moving from global to local structure to solve the graph-matching problem. *Pattern Recogn. Lett.* **65**, 204–210 (2015)
4. Caetano, T.S., McAuley, J.J., Cheng, L., Le, Q.V., Smola, A.J.: Learning graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(6), 1048–1058 (2009)
5. Cortés, X., Serratos, F.: Learning graph matching substitution weights based on the ground truth node correspondence. *IJPRAI* **30**(2) (2016)
6. Schmidhuber, J.: Deep learning in neural networks: an overview. *Neural Netw.* **61**, 85–117 (2015)
7. Raveaux, R., Martineau, M., Conte, D., Venturini, G.: Learning graph matching with a graph-based perceptron in a classification context. In: Foggia, P., Liu, C.-L., Vento, M. (eds.) *GbRPR 2017*. LNCS, vol. 10310, pp. 49–58. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-58961-9\\_5](https://doi.org/10.1007/978-3-319-58961-9_5)
8. Neuhaus, M., Bunke, H.: Self-organizing maps for learning the edit costs in graph matching. *IEEE Trans. Syst. Man Cybern. Part B* **35**(3), 503–514 (2005)
9. Neuhaus, M., Bunke, H.: Automatic learning of cost functions for graph edit distance. *Inf. Sci.* **177**(1), 239–247 (2007)
10. Leordeanu, M., Sukthankar, R., Hebert, M.: Unsupervised learning for graph matching. *Int. J. Comput. Vis.* **96**(1), 28–45 (2012)
11. Serratos, F., Solé-Ribalta, A., Cortés, X.: Automatic learning of edit costs based on interactive and adaptive graph recognition. In: Jiang, X., Ferrer, M., Torsello, A. (eds.) *GbRPR 2011*. LNCS, vol. 6658, pp. 152–163. Springer, Heidelberg (2011). [https://doi.org/10.1007/978-3-642-20844-7\\_16](https://doi.org/10.1007/978-3-642-20844-7_16)
12. Cortés, X., Serratos, F.: Learning graph-matching edit-costs based on the optimality of the oracle’s node correspondences. *Pattern Recogn. Lett.* **56**, 22–29 (2015)

13. Riesen, K., Ferrer, M.: Predicting the correctness of node assignments in bipartite graph matching. *Pattern Recogn. Lett.* **69**, 8–14 (2016)
14. Kanzow, C., Yamashita, N., Fukushima, M.: Levenberg-Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints. *JCAM* **172**(2), 375–397 (2004)
15. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**(4), 950–959 (2009)
16. Kuhn, H.W.: The Hungarian method for the assignment problem. *Naval Res. Log. Q.* **2**, 83–97 (1955)
17. Moreno-García, C.F., Cortés, X., Serratosa, F.: A graph repository for learning error-tolerant graph matching. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) *S+SSPR 2016*. LNCS, vol. 10029, pp. 519–529. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_46](https://doi.org/10.1007/978-3-319-49055-7_46)



# Error-Tolerant Geometric Graph Similarity

Shri Prakash Dwivedi<sup>(✉)</sup> and Ravi Shankar Singh

Department of Computer Science and Engineering,  
Indian Institute of Technology (BHU), Varanasi, India  
{shripd.rs.cse16,ravi.cse}@iitbhu.ac.in

**Abstract.** Graph matching is the task of computing the similarity between two graphs. Error-tolerant graph matching is a type of graph matching, in which a similarity between two graphs is computed based on some tolerance value whereas within exact graph matching a strict one-to-one correspondence is required between two graphs. In this paper, we present an approach to error-tolerant graph similarity using geometric graphs. We define the vertex distance (dissimilarity) and edge distance between two graphs and combine them to compute graph distance.

**Keywords:** Graph matching · Geometric graph · Graph distance

## 1 Introduction

Computing the similarity between two graphs is one of the fundamental problems of computer science. Graph Matching (GM) is the process of finding similarity between two graphs. It has become one of the engaging areas of research over the last few decades. The major GM applications include structural pattern recognition, computer vision, biometrics, chemical and biological applications, etc. GM is usually classified into two types which are known as exact GM and inexact or error-tolerant GM. Exact GM is like graph isomorphism problem, where a bijective mapping is required from the nodes of the first graph to the nodes of the second graph such that if there is an edge in the first graph connecting two nodes, then there exists an edge in the second graph connecting the corresponding set of nodes.

Error-tolerant GM provides a flexible approach towards GM problem as opposed to exact GM which performs a strict matching. In many practical applications, the input data get modified due to the presence of noise and therefore exact GM may not be suitable [6]. For such kind of applications, error-tolerant GM offers the tolerance to noise by computing a similarity score between two graphs.

The optimal solution to exact GM problem takes exponential time as a function of the number of nodes in input graph. The complexity of graph isomorphism problem is neither known to be in  $NP$ -complete nor in  $P$ , whereas subgraph isomorphism is known to be in class  $NP$ -complete. Since exact polynomial time

algorithms for GM problem is not available, several suboptimal solutions to GM problem have been proposed in the literature.

An extensive survey of various GM methods is given in [6,8]. In [2] author describes a precise framework for error-tolerant GM.  $A^*$  search technique for finding minimum cost paths is described in [10]. Error-tolerant GM for the attributed relational graph (ARG) is described in [26]. In [21] authors specify a distance measure for ARG by considering the cost of recognition of nodes.

A class of GM algorithms using spectral method is described in [4,17,24]. The spectral technique relies on the fact that the adjacency matrix of a graph does not change on node rearrangement accordingly adjacency matrix will have equivalent eigendecomposition for similar graphs.

A novel class of GM methods utilizing graph kernel is described in [9,15]. Kernel methods enable us to apply statistical pattern recognition techniques to graph domain. The major types of graph kernel include convolution kernel, diffusion kernel and random walk kernel [11,13].

Graph Edit Distance (GED) is one of the most widely used method for error-tolerant GM [3,21]. GED between two graphs is defined as the minimum number of edit operations needed to transform the first graph into another one. GED is the generalization of string edit distance. Exact algorithms for GED are computationally expensive and is exponential on the size of input graphs. In order to make GED computation more feasible, many approximation techniques based on local search, greedy approach, neighborhood search, bipartite GED etc. have been proposed [7,14,19,20,25].

Another class of GM methods is based on geometric graphs in which every vertex has an associated coordinate in two-dimensional space. In [12] authors have shown that geometric graph isomorphism can be performed in polynomial time. Geometric GM using edit distance approach is demonstrated to be  $NP$ -hard in [5]. Geometric GM using probabilistic approach is described in [1] and in the paper, [16] authors have presented geometric GM based on Monte Carlo tree search. In [23] authors defines spectral graph distance using the difference between the spectra of the Laplacian matrices of the two graphs. In [22] authors introduced a method for network comparison that can quantify topological differences between networks.

The geometric graph is a graph in which each vertex has a unique coordinate point. Due to this additional information, geometric graphs may offer an alternative approach to traditional GM techniques. In this paper, we propose an approach to error-tolerant graph similarity for geometric graphs. We define the vertex distance between two geometric graphs as the minimum of the sum of the Euclidean distances between the corresponding coordinates from one geometric graph to another one. We define edge distance by representing each edge of a geometric graph using two parameters, its angular orientation from positive  $x$ -axis and its length. Finally, we integrate both vertex distance and edge distance to compute a measure of similarity between two geometric graphs.

This paper is organized as follows. Section 2, contains basic definitions and notation. Section 3, defines vertex distance, edge distance and algorithm to

compute the graph distance between two graphs. Section 4, describes results with discussion and finally Sect. 5, contains the conclusion.

## 2 Basic Concepts and Notation

In this section, we review the basic definitions and notations used in exact and error-tolerant GM.

A graph  $g$  is defined as  $g = (V, E, \mu, \nu)$ , where  $V$  is the set of vertices,  $E$  is the set of edges,  $\mu : V \rightarrow L_V$  is a mapping that allocates a vertex label alphabet  $l \in L_V$  to each vertex  $v \in V$ ,  $\nu : E \rightarrow L_E$  is a mapping that allocates an edge label alphabet  $l_e \in L_E$  to every edge in  $E$ . Where,  $L_V$  and  $L_E$  are vertex label set and edge label set respectively. If  $L_V = L_E = \emptyset$  then  $g$  is called the unlabeled graph.

A graph  $g_1$  is said to be a subgraph of graph  $g_2$ , if  $V_1 \subseteq V_2$ ;  $E_1 \subseteq E_2$ ; for every node  $u \in g_1$ , we have  $\mu_1(u) = \mu_2(u)$ ; similarly, for every edge  $e \in g_1$ , we have  $\nu_1(e) = \nu_2(e)$ .

A graph isomorphism between two graphs  $g_1$  and  $g_2$  is defined as a bijective mapping between every vertex  $u \in g_1$  to a unique vertex  $v \in g_2$ , such that their labels and edges are preserved.

Let  $g_1$  and  $g_2$  be two graphs. A function  $f : V_1 \rightarrow V_2$  from  $g_1$  to  $g_2$  is called as subgraph isomorphism if there is a graph isomorphism between  $g_1$  and a subgraph of  $g_2$ .

Let  $g_1$  and  $g_2$  be two graphs. A one-to-one correspondence function  $f : V'_1 \rightarrow V'_2$  from  $g_1$  to  $g_2$  is called an error-tolerant GM, if  $V'_1 \subseteq V_1$  and  $V'_2 \subseteq V_2$  [2].

A geometric graph  $G$  is defined as  $G = (V, E, l, c)$ , where  $V$  is the set of vertices,  $E$  is the set of edges,  $l$  is a labeling function  $l : \{V \cup E\} \rightarrow \Sigma$  which assigns a label from  $\Sigma$  to each vertex and edge,  $c$  is a function  $c : V \rightarrow \mathbb{R}^2$  which assigns a coordinate point to each vertex of  $G$ . If  $\Sigma = \emptyset$  then  $G$  is called the unlabeled geometric graph.

## 3 Geometric Graph Similarity

In this section, we introduce vertex distance and edge distance between the geometric graphs  $G_1$  and  $G_2$ . We use these distance measures to compute the dissimilarity or graph distance between two graphs.

**Definition 1.** Let  $G_1 = (V_1, E_1, l_1, c_1)$  and  $G_2 = (V_2, E_2, l_2, c_2)$  be two geometric graphs with  $|V_1| = |V_2| = n$ . Let coordinate points of  $V_1$  be  $\{(a_1, b_1), (a_2, b_2), \dots, (a_n, b_n)\}$  and coordinate points of  $V_2$  be  $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$  then the vertex distance or dissimilarity between the two graphs  $G_1$  and  $G_2$  is defined as

$$VD(G_1, G_2) = \min_{1 \leq i, j \leq n} \sum \sqrt{(a_i - x_j)^2 + (b_i - y_j)^2} \tag{1}$$

Here,  $VD$  represents the minimum sum of the distance of each pair of assigned vertices from  $V_1$  to  $V_2$ . Larger deviation of corresponding coordinates between  $G_1$  and  $G_2$  implies a larger  $VD$  value.

We can show that  $VD(G_1, G_2)$  is a metric. Here  $VD(G_1, G_2) \geq 0$ . if  $G_1 = G_2$  then  $VD(G_1, G_2) = 0$ , and if  $VD(G_1, G_2) = 0$  then  $\min_{1 \leq i, j \leq n} \sum [(a_i - x_j)^2 + (b_i - y_j)^2]^{1/2} = 0$ , which implies that each individual sum of this expression is 0 and therefore  $G_1 = G_2$ . Also  $VD(G_1, G_2) = VD(G_2, G_1)$ , therefore it is symmetric, and finally  $VD(G_1, G_2) \leq VD(G_1, G_3) + VD(G_3, G_2)$  follows from the Euclidean distance property  $d(x, y) \leq d(x, z) + d(z, y)$ .

For a geometric graph  $G_1$ , let  $|V_1| = n$ . Then the  $n \times n$  adjacency matrix  $A = (a_{ij})_{n \times n}$  of  $G_1$  can be defined by

$$a_{ij} = \begin{cases} \{(a_i, b_i), (a_j, b_j)\}, & \text{if } \{(a_i, b_i), (a_j, b_j)\} \in E_1 \\ \varepsilon, & \text{otherwise} \end{cases}$$

Similarly, the  $n \times n$  adjacency matrix  $A = (x_{ij})_{n \times n}$  of  $G_2$  can be defined by

$$x_{ij} = \begin{cases} \{(x_i, y_i), (x_j, y_j)\}, & \text{if } \{(x_i, y_i), (x_j, y_j)\} \in E_2 \\ \varepsilon, & \text{otherwise} \end{cases}$$

Let  $\theta_{\{(a,b),(c,d)\}}$  denote the angle subtended between the line joining the coordinate points  $(a, b)$ ,  $(c, d)$  and positive  $x$ -axis.

**Definition 2.** Let  $G_1 = (V_1, E_1, l_1, c_1)$  and  $G_2 = (V_2, E_2, l_2, c_2)$  be two geometric graphs with  $|V_1| = |V_2| = n$ . Then the edge distance or dissimilarity between the two graphs  $G_1$  and  $G_2$  is defined as

$$ED(G_1, G_2) = \min_{1 \leq i, j \leq n} \sum \left( \sqrt{\left( (\Theta_{ij} - \Theta'_{ij}) \frac{\pi}{180^\circ} \right)^2} + \sqrt{(d_{ij} - D_{ij})^2} \right) \quad (2)$$

where,  $\Theta_{ij} = \theta_{\{(a_i, b_i), (a_j, b_j)\}}$ ,  $\Theta'_{ij} = \theta_{\{(x_i, y_i), (x_j, y_j)\}}$ ,  $d_{ij} = \sqrt{(a_i - a_j)^2 + (b_i - b_j)^2}$ , and  $D_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ .

The first term in the above definition of  $ED$  accounts for the angular distance in radian between each pair of corresponding edges selected from  $E_1$  and  $E_2$ , whereas the second term of  $ED$  represents the difference of edge length between each pair of assigned edges. Similar to  $VD$ , we can show that  $ED(G_1, G_2) \geq 0$ . If  $G_1 = G_2$  then  $ED(G_1, G_2) = 0$ . But when  $ED(G_1, G_2) = 0$  then  $G_1$  is not necessarily equal to  $G_2$ . We can observe that  $ED$  between two translated or rotated version of same geometric graph remains 0. Also,  $ED$  follows triangle inequality since both first and second term of  $ED$  follows triangle inequality property.

### 3.1 Graph Distance Algorithm

The computation of graph distance between two geometric graphs  $G_1$  and  $G_2$  is described in Algorithm 1. The input to the algorithm is two geometric graphs

$G_1$  and  $G_2$  and three weighting parameters  $w_1, w_2$  and  $w_3$ , which are application dependent. By default we take equal weighting factors, that is  $w_1 = w_2 = w_3$ . The output of the algorithm is graph distance between  $G_1$  and  $G_2$ . One optional step of this algorithm is preprocessing of input graphs. If one graph is identical to other by performing geometric transformation like translation, rotation, and scaling, then the input graphs are processed to make their coordinate reference frame aligned. Line 1 of the algorithm computes the assignment of vertices from  $V_1$  to  $V_2$  based on their coordinate such that  $VD$  is minimum. We can use the Munkres algorithm for optimal assignment of vertices, or we can start with the lowest  $x$ -coordinate of the vertex from  $V_1$  and assign it to the nearest vertex from  $V_2$  and so on. Similarly, assignment of edges from  $E_1$  to  $E_2$  is performed in line 2. Vertex distance  $VD$  is evaluated in line 3, and edge distance is computed in lines 3–4. Whereas  $ED_1$  consists of the difference of angular distance between two assigned edges, on the other hand,  $ED_2$  contains difference of Euclidean distance between two assigned edges. Finally, graph distance is computed in line 6, using the weighting factors  $w_1, w_2$ , and  $w_3$ .

---

**Algorithm 1. Graph-Distance** ( $G_1, G_2, w_1, w_2, w_3$ )

---

**Require:** Two undirected unlabeled geometric graphs  $G_1, G_2$ , where  $G_i = (V_i, E_i, c_i)$  for  $i = 1, 2$ , and weighting factors  $w_i$  for  $i = 1$  to 3

**Ensure:** Graph distance or dissimilarity value between  $G_1$  and  $G_2$   
 ▷ preprocessing of input graphs  $G_1$  and  $G_2$

- 1: Compute vertex assignment from  $V_1$  to  $V_2$
  - 2: Compute edge assignment from  $E_1$  to  $E_2$
  - 3:  $VD \leftarrow \sum_{i,j=1}^n \sqrt{(a_i - x_j)^2 + (b_i - y_j)^2}$
  - 4:  $ED_1 \leftarrow \sum_{i,j=1}^n \left( \sqrt{\left( (\Theta_{ij} - \Theta'_{ij}) \frac{\pi}{180^\circ} \right)^2} \right)$
  - 5:  $ED_2 \leftarrow \sum_{i,j=1}^n \sqrt{(d_{ij} - D_{ij})^2}$
  - 6:  $GD \leftarrow w_1 \cdot VD + w_2 \cdot ED_1 + w_3 \cdot ED_2$
  - 7: **return** ( $GD$ )
- 

**Proposition 1.** *Graph-Distance algorithm executes in  $O(n^3)$  time.*

We can observe that the assignment of vertices and edges in lines 1–2 can be performed in  $O(n^3)$  by Munkres algorithm and the remaining steps can be computed in  $O(n^2)$ ; therefore overall execution time remains  $O(n^3)$ .

## 4 Results and Discussion

The proposed graph distance measure can be used to compare the structural similarity between different graphs. In the definition of vertex distance and edge distance, we have assumed that  $|V_1| = |V_2|$  this limitation can be resolved by adding extra vertices with  $(0, 0)$  coordinate to the smaller vertex set so that the size of the graph becomes equal. A more reasonable option is to use coordinates

with the mean value for  $x$  and  $y$  in the smaller graph. That is, if  $|V_1| = m$  and  $|V_2| = n$  where  $m > n$  then  $(m - n)$  vertices of  $G_2$  are allocated the coordinates  $(x_{mean}, y_{mean})$  in the preprocessing step of the Graph-Distance algorithm. Here  $x_{mean}$  and  $y_{mean}$  are the mean of  $x$  and  $y$  values of coordinates of  $n$  vertices of  $G_2$ .

In order to compare graph distance computed using Graph-Distance algorithm and GED computed using  $A^*$  algorithm we use Letter dataset of IAM graph database repository [18]. Letter dataset consists of graph representing capital letters of alphabets, drawn using straight lines only. Distortions of three different levels are applied to prototype graphs to produce three classes of Letter dataset, which are high, medium and low. Letter graphs in high class are more deformed than that of graph is medium or low class. Table 1 shows the comparison of graph distance with GED computed between the first graph and next 10 graphs of each three classes of Letter dataset.  $GD_{HIGH}$ ,  $GD_{MED}$  and  $GD_{LOW}$  in this table represents Graph-Distance computed for graphs of high, medium and low classes respectively. Similarly,  $GED_{HIGH}$ ,  $GED_{MED}$  and  $GED_{LOW}$  denotes GED computed for graphs of high, medium and low classes respectively. In this table, we observe that largest graph distance under  $GD_{HIGH}$  also corresponds to largest GED under  $GED_{HIGH}$ , whereas the smallest graph distance under  $GD_{HIGH}$  corresponds to second smallest GED under  $GED_{HIGH}$ . One advantage of distance computed using Graph-Distance algorithm is that it is symmetric, on the other hand, GED may not be symmetric. Another advantage is that Graph-Distance algorithm is efficient and it can process the graph having even more than 100 nodes, whereas GED may not be executed on graphs having more than 10–20 nodes.

**Table 1.** Graph distance vs Graph edit distance

$GD_{HIGH}$	$GED_{HIGH}$	$GD_{MED}$	$GED_{MED}$	$GD_{LOW}$	$GED_{LOW}$
7.061	3.152	7.267	2.307	4.643	1.285
6.347	3.050	10.347	3.056	7.186	2.293
4.551	2.111	7.131	3.433	5.275	1.387
5.669	3.092	12.015	2.843	5.163	1.358
8.926	3.067	10.048	4.061	6.066	2.458
12.251	4.148	6.971	2.371	4.891	1.317
5.651	2.808	7.457	2.402	5.430	1.339
5.588	2.342	7.563	3.830	5.862	2.336
4.114	2.318	6.753	3.528	4.827	1.036
6.414	2.238	5.582	2.025	3.486	1.778

Geometric graph similarity can be particularly useful in real-world applications, where the graph data is large and can be modified by noise or distortions. Depending on application requirement, we can select weighting factors such that



$\sum_{i=1}^3 w_i = 1$ . In the above experiment we used equal weighting parameters, i.e.,  $w_1 = w_2 = w_3 = 1/3$ . When the position of vertices is more dominant, we can select  $w_1$  to be higher, if angular structures are more important then  $w_2$  can be prominent. Otherwise, if edge differences are more essential, we can select  $w_3$  to be higher.

## 5 Conclusion

In this paper, we described an approach to compute inexact geometric graph distance between two graphs. In a geometric graph, every vertex has an associated coordinate, which specify its distinct position in the plane. We can use this fact to define the distance between two graphs. First, we introduced vertex dissimilarity between two geometric graphs. Then we defined edge dissimilarity between two geometric graphs. Then we used them to find the similarity between two graphs. Also, we applied the graph distance similarity measure to some Letter graphs and observed some of its advantages.

## References

1. Armiti, A., Gertz, M.: Geometric graph matching and similarity: a probabilistic approach. In: SSDBM (2014)
2. Bunke, H.: Error-tolerant graph matching: a formal framework and algorithms. In: Amin, A., Dori, D., Pudil, P., Freeman, H. (eds.) SSPR/SPR 1998. LNCS, vol. 1451, pp. 1–14. Springer, Heidelberg (1998). <https://doi.org/10.1007/BFb0033223>
3. Bunke, H., Allerman, G.: Inexact graph matching for structural pattern recognition. *Pattern Recogn. Lett.* **1**, 245–253 (1983)
4. Caelli, T., Kosinov, S.: Inexact graph matching using eigen-subspace projection clustering. *Int. J. Pattern Recogn. Artif. Intell.* **18**(3), 329–355 (2004)
5. Cheong, O., Gudmundsson, J., Kim, H.-S., Schymura, D., Stehn, F.: Measuring the similarity of geometric graphs. In: Vahrenhold, J. (ed.) SEA 2009. LNCS, vol. 5526, pp. 101–112. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-02011-7\\_11](https://doi.org/10.1007/978-3-642-02011-7_11)
6. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. *Int. J. Pattern Recogn. Artif. Intell.* **18**(3), 265–298 (2004)
7. Dwivedi, S.P., Singh, R.S.: Error-tolerant graph matching using homeomorphism. In: International Conference on Advances in Computing, Communication and Informatics (ICACCI), pp. 1762–1766 (2017)
8. Foggia, P., Percannella, G., Vento, M.: Graph matching and learning in pattern recognition in the last 10 years. *Int. J. Pattern Recogn. Artif. Intell.* **88**, 1450001.1–1450001.40 (2014)
9. Gartner, T.: *Kernels for Structured Data*. World Scientific, Singapore (2008)
10. Hart, P.E., Nilson, N.J., Raphael, B.: A formal basis for heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **4**, 100–107 (1968)
11. Haussler, D.: Convolution kernels on discrete structures. Technical report, UCSC-CRL-99-10, University of California, Sant Cruz (1999)
12. Kuramochi, M., Karypis, G.: Discovering frequent geometric subgraphs. *Inf. Syst.* **32**, 1101–1120 (2007)

13. Lafferty, J., Lebanon, G.: Diffusion kernels on statistical manifolds. *J. Mach. Learn. Res.* **6**, 129–163 (2005)
14. Neuhaus, M., Riesen, K., Bunke, H.: Fast suboptimal algorithms for the computation of graph edit distance. In: Yeung, D.-Y., Kwok, J.T., Fred, A., Roli, F., de Ridder, D. (eds.) *SSPR /SPR 2006*. LNCS, vol. 4109, pp. 163–172. Springer, Heidelberg (2006). [https://doi.org/10.1007/11815921\\_17](https://doi.org/10.1007/11815921_17)
15. Neuhaus, M., Bunke, H.: *Bridging the Gap Between Graph Edit Distance and Kernel Machines*. World Scientific, Singapore (2007)
16. Pinheiro, M.A., Kybic, J., Fua, P.: Geometric graph matching using Monte Carlo tree search. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(11), 2171–2185 (2017)
17. Robles-Kelly, A., Hancock, E.R.: Graph edit distance from spectral seriation. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**, 365–378 (2005)
18. Riesen, K., Bunke, H.: IAM graph database repository for graph based pattern recognition and machine learning. In: da Vitoria Lobo, N., et al. (eds.) *SSPR /SPR 2008*. LNCS, vol. 5342, pp. 287–297. Springer, Berlin (2008). [https://doi.org/10.1007/978-3-540-89689-0\\_33](https://doi.org/10.1007/978-3-540-89689-0_33)
19. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**(4), 950–959 (2009)
20. Riesen, K., Bunke, H.: Improving bipartite graph edit distance approximation using various search strategies. *Pattern Recogn.* **48**(4), 1349–1363 (2015)
21. Sanfeliu, A., Fu, K.S.: A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man Cybern.* **13**(3), 353–363 (1983)
22. Schieber, T.A., Carpi, L., Diaz-Guilera, A., Pardalos, P.M., Masoller, C., Ravetti, M.G.: Quantification of network structural dissimilarities. *Nature Commun.* **8**(13928), 1–10 (2017)
23. Shimada, Y., Hirata, Y., Ikeguchi, T., Aihara, K.: Graph distance for complex networks. *Sci. Rep.* **6**(34944), 1–6 (2016)
24. Shokoufandeh, A., Macrini, D., Dickinson, S., Siddiqi, K., Zucker, S.: Indexing hierarchical structures using graph spectra. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(3), 365–378 (2005)
25. Sorlin, S., Solnon, C.: Reactive tabu search for measuring graph similarity. In: Brun, L., Vento, M. (eds.) *GbrPR 2005*. LNCS, vol. 3434, pp. 172–182. Springer, Heidelberg (2005). [https://doi.org/10.1007/978-3-540-31988-7\\_16](https://doi.org/10.1007/978-3-540-31988-7_16)
26. Tsai, W.H., Fu, K.S.: Error-correcting isomorphisms of attributed relational graphs for pattern analysis. *IEEE Trans. Syst. Man Cybern.* **9**, 757–768 (1979)



# Learning Cost Functions for Graph Matching

Rafael de O. Werneck<sup>1</sup>(✉), Romain Raveaux<sup>2</sup>, Salvatore Tabbone<sup>3</sup>,  
and Ricardo da S. Torres<sup>1</sup>

<sup>1</sup> Institute of Computing, University of Campinas, Campinas, SP, Brazil  
{rafael.werneck,rtorres}@ic.unicamp.br

<sup>2</sup> Université François Rabelais de Tours, 37200 Tours, France  
romain.raveaux@univ-tours.fr

<sup>3</sup> Université de Lorraine-LORIA UMR 7503, Vandoeuvre-lès-Nancy, France  
tabbone@loria.fr

**Abstract.** During the last decade, several approaches have been proposed to address detection and recognition problems, by using graphs to represent the content of images. Graph comparison is a key task in those approaches and usually is performed by means of graph matching techniques, which aim to find correspondences between elements of graphs. Graph matching algorithms are highly influenced by cost functions between nodes or edges. In this perspective, we propose an original approach to learn the matching cost functions between graphs' nodes. Our method is based on the combination of distance vectors associated with node signatures and an SVM classifier, which is used to learn discriminative node dissimilarities. Experimental results on different datasets compared to a learning-free method are promising.

**Keywords:** Graph matching · Cost learning · SVM

## 1 Introduction

In the pattern recognition domain, we can represent objects using two methods: statistical or structural [4]. On the later, objects are represented by a data structure (*e.g.*, graphs, trees), which encodes their components and relationships; and on the former, objects are represented by means of feature vectors. Most methods for classification and retrieval in the literature are limited to statistical representations [17]. However, structural representation are more powerful, as the object components and their relations are described in a single formalism [18]. Graphs are one of the most used structural representations. Unfortunately, graph

---

R. de O. Werneck—Thanks to CNPq (grant #307560/2016-3), CAPES (grant #88881.145912/2017-01), FAPESP (grants #2016/18429-1, #2017/16453-5, #2014/12236-1, #2015/24494-8, #2016/50250-1, and #2017/20945-0), and the FAPESP-Microsoft Virtual Institute (#2013/50155-0, #2013/50169-1, and #2014/50715-9) agencies for funding.

comparison suffers from high complexity, often an NP-hard problem requiring exponential time and space to find the optimal solution [5].

One of the widely used method for graph matching is the graph edit distance (GED). GED is an error-tolerant graph matching paradigm that defines the similarity of two graphs by the minimum number of edit operations necessary to transform one graph into another [3]. A sequence of edit operations that transforms one graph into another is called edit path between two graphs. To quantify the modifications implied by an edit path, a cost function is defined to measure the changes proposed by each edit operation. Consequently, we can define the edit distance between graphs as the edit path with minimum cost.

The possible edit operations are: node substitution, edge substitution, node deletion, edge deletion, node insertion, and edge insertion. The cost function is of first interest and can change the problem being solved. In [1,2], a particular cost function for the GED is introduced, and it was shown that under this cost function, the GED computation is equivalent to the maximum common sub-graph problem. Neuhaus and Bunke [14], in turn, showed that if each elementary operation satisfies the criteria of a metric distance (separability, symmetry, and triangular inequality) then the GED is also a metric.

Usually, cost functions are manually designed and are domain-dependent. Domain-dependent cost functions can be tuned by learning weights associated with them. In Table 1, published papers dealing with edit cost learning are tabulated. Two criteria are optimized in the literature, the matching accuracy between graph pairs or an error rate on a classification task (classification level). In [13], learning schemes are applied on the GED problem while in [6,11], other matching problems are addressed. In [11], the learning strategy is unsupervised as the ground truth is not available. In another research venue, different optimization algorithms are used. In [12], Self-Organizing Maps (SOMs) are used to cluster substitution costs in such a way that the node similarity of graphs from the same class is increased, whereas the node similarity of graphs from different classes is decreased. In [13], Expectation Maximization algorithm (EM) is used for the same purpose. An assumption is made on attribute types. In [7], the learning problem is mapped to a regression problem and a structured support vector machine (SSVM) is used to minimize it. In [8], a method to learn scalar values for the insertion and deletion costs on nodes and edges is proposed. An extension to substitution costs is presented in [9]. The contribution presented in [16] is the nearest work to our proposal. In that work, the node assignment is represented as a vector of 24 features. These numerical features are extracted from a node-to-node cost matrix that is used for the original matching process. Then, the assignments derived from exact graph edit distance computation is used as ground truth. On this basis, each node assignment computed is labeled as correct or incorrect. This set of labeled assignments is used to train an SVM endowed with a Gaussian kernel in order to classify the assignments computed by the approximation as correct or incorrect. This work operates at the matching level. All prior works rely on predefined cost functions adapted to fit an objective of matching accuracy. Little research has been carried out to automatically design generic cost functions in a classification context.

**Table 1.** Graph matching learning approaches.

Ref.	Graph matching problem	Supervised	Criterion	Optimization method
[12]	GED	Yes	Recognition rate	SOM
[13]	GED	Yes	Recognition rate	EM
[8,9]	GED	Yes	Matching accuracy	Quadratic programming
[6]	Other	Yes	Matching accuracy	Bundle
[7]	Other	Yes	Matching accuracy	SSVM
[11]	Other	No	Matching accuracy	Bundle

In this paper, we propose to learn a discriminative cost function between nodes with no restriction on graph types nor on labels for a classification task. On a training set of graphs, a feature vector is extracted from each node of each graph thanks to a node signature that describes local information in graphs. Node dissimilarity vectors are obtained by pairwise comparison of the feature vectors. Node dissimilarity vectors are labeled according to the node pair belonging to graphs of the same class or not. On this basis, an SVM classifier is trained. At the decision stage, two graphs are compared, a new node pair is given as an input of the classifier, and the class membership probability is outputted. These adapted costs are used to fill a node-to-node similarity matrix. Based on these learned matching costs, we approximate the matching graph problem as a Linear Sum Assignment Problem (LSAP) between the nodes of two graphs. The LSAP aims at finding the maximum weight matching between the elements of two sets and this problem can be solved by the Hungarian algorithm [10] in  $O(n^3)$  time.

The paper is organized as follow: Sect. 2 presents our approach for local description of graphs, and the proposed approaches to populate the cost matrix for the Hungarian algorithm. Section 3 details the datasets and the adopted experimental protocol, as well as presents the results and discussions about them. Finally, Sect. 4 is devoted to our conclusions and perspectives for future work.

## 2 Proposed Approach

In this section, we present our proposal to resolve the graph matching problem as a bipartite graph matching using local information.

### 2.1 Local Description

In this work, we use node signatures to obtain local descriptions of graphs. In order to define the signature, we use all information of the graph and the node. Our node signature is represented by the node attributes, node degree, attributes of incident edges, and degrees of the nodes connected to the edges.

Given a general graph  $G = (V, E)$ , we can define the node signature extraction process and representation, respectively, as:

$$\begin{aligned} \Gamma(G) &= \{\gamma(n) | \forall n \in V\} \\ \gamma(n) &= \{\alpha_n^G, \theta_n^G, \Delta_n^G, \Omega_n^G\} \end{aligned}$$

where  $\alpha_n^G$  is the attributes of the node  $n$ ,  $\theta_n^G$  is the degree of node  $n$ ,  $\Delta_n^G$  is the set of degrees of adjacent nodes to  $n$ , and  $\Omega_n^G$  is a set of attributes of the incident edges of  $n$ .

## 2.2 HEOM Distance

One of our approaches to perform graph matching consists on finding the minimum distance to transform the node signatures from one graph into the node signatures from another graph. To calculate the distance between two node signatures, we need a distance metric capable of dealing with numeric and symbolic attributes. We selected the *Heterogeneous Euclidean Overlap Metric* (HEOM) [19] and we provided an adaptation for our graph local description.

The HEOM distance is defined as:

$$HEOM(i, j) = \sqrt{\sum_{a=0}^n \delta(i_a, j_a)^2}, \tag{1}$$

where  $a$  is each attribute of the vector, and  $\delta(i_a, j_a)$  is defined as:

$$\delta(i_a, j_a) = \begin{cases} 1 & \text{if } i_a \text{ or } j_a \text{ is missing,} \\ 0 & \text{if } a \text{ is symbolic and } i_a = j_a, \\ 1 & \text{if } a \text{ is symbolic and } i_a \neq j_a, \\ \frac{|i_a - j_a|}{\text{range}_a} & \text{if } a \text{ is numeric.} \end{cases} \tag{2}$$

In our approach, we define the distance between two node signatures as follow. Let  $A = (V_a, E_a)$  and  $B = (V_b, E_b)$  be two graphs and  $n_a \in V_a$  and  $n_b \in V_b$  be two nodes from these graphs. Let  $\gamma(n_a)$  and  $\gamma(n_b)$  be the signature of these nodes, that is:

$$\gamma(n_a) = \{\alpha_{n_a}^A, \theta_{n_a}^A, \Delta_{n_a}^A, \Omega_{n_a}^A\}$$

and

$$\gamma(n_b) = \{\alpha_{n_b}^B, \theta_{n_b}^B, \Delta_{n_b}^B, \Omega_{n_b}^B\}.$$

The distance  $\epsilon$  between two node signatures is:

$$\begin{aligned} \epsilon(\gamma(n_a), \gamma(n_b)) &= HEOM(\alpha_{n_a}^A, \alpha_{n_b}^B) + HEOM(\theta_{n_a}^A, \theta_{n_b}^B) \\ &+ HEOM(\Delta_{n_a}^A, \Delta_{n_b}^B) + \frac{\sum_{i=1}^{|\Omega_{n_a}^A|} HEOM(\Omega_{n_a}^A(i), \Omega_{n_b}^B(i))}{|\Omega_{n_a}^A|} \end{aligned} \tag{3}$$

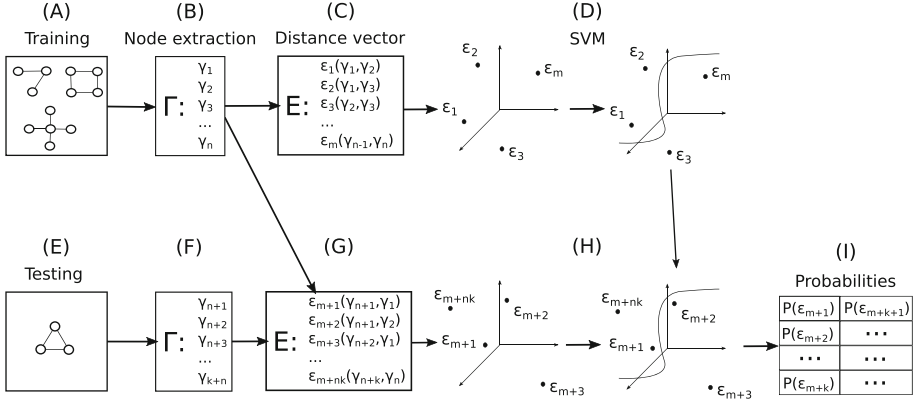


Fig. 1. Proposed SVM approach to compute the edit cost matrix.

### 2.3 SVM-Based Node Dissimilarity Learning

We propose an SVM approach to learn the graph edit distance between two graphs. In this approach, we first define a *distance vector*  $\epsilon'$  between two node signatures. Function  $\epsilon'$  is derived from  $\epsilon$ , but instead of summing up the distance related to all structures, the function considers each structure distance score as a value of a bin of the vector. This distance vector is composed of the HEOM distance between each structure of the node signature, i.e., the distance between the node attribute, node degree, degrees of the nodes connected to the edges, and attributes of incident edges are components of the vector, i.e.,

$$\epsilon'(\gamma(n_a), \gamma(n_b)) = [HEOM(\gamma(n_a)_i, \gamma(n_b)_i)],$$

$$\forall i \in \{0, \dots, |\gamma(n)|\} \mid \gamma(n)_i \text{ is a component of } \gamma(n).$$

To each distance vector  $\epsilon'$ , a label is assigned. These labels guide the SVM learning process. We propose the following formulation to assign labels to distance vectors. Let  $Y = \{y_1, y_2, \dots, y_l\}$  be the set of  $l$  labels associated with graphs. In our formulation, denominated multi-class, distance vectors, which are associated with node signatures extracted from graphs of the same class (say  $y_i$ ), are labeled as  $y_i$ . Otherwise, a novel label  $y_{l+1}$  is used, representing that the distance vectors were computed from node signatures belonging to graphs belonging to different classes.

Figure 1 illustrates the main steps of our approach. Given a set of training graphs (step A in the figure), we first extract the node signatures from all graphs (B), and compute the pairwise distance vectors (C). We then use the labeling procedure described above to assign labels to distance vectors defined by node signatures extracted from graphs of the training set and use these labeled vectors to train an SVM classifier (D).

## 2.4 Graph Classification

At testing stage, each one of the graphs from the test set (E) has its node signatures extracted (F). Again, distance vectors are computed, now considering node signatures from the test and from the training set (G). With the distance vectors, we can project them into the learned feature space and obtain the probability of a test sample that belongs to the training set classes considering the SVM hyperplane of separation (H). These probabilities are used to populate a cost matrix for each graph in the training set (I), in such a way that, for each node signature from the test graph (row) and each node signature from the training graph (column), we create a matrix of probabilities for each combination of test and training graphs. This matrix is later used in the Hungarian algorithm. As the resulting cost matrices encodes probabilities, we compute the maximum cost path using the Hungarian algorithm instead of the minimum. The test sample classification is based on the k-nearest neighbor (kNN) graphs found in the training set, where graph similarity is defined by the Hungarian algorithm.

## 3 Experimental Results

In this section, we describe the datasets used in the experiments, we present our experimental protocol, and how our method was evaluated. At the end, we present our results and discuss them.

### 3.1 Datasets

In our paper, we perform experiments in three labeled datasets from the IAM graph database [15]: Letter, Mutagenicity, and GREC.

The **Letter** database comprises 15 classes of distorted letter drawings. Each letter is represented by a graph, in which the nodes are ending points of lines, and edges are the lines connecting ending points. The attributes of the node are its position. This dataset has three sub-datasets, considering different distortions (low distortion, medium distortion, and a high distortion).

**Mutagenicity** is a database of 2 classes representing molecular compounds. In this database, the nodes are the atoms and the edges the valence of the linkage.

**GREC** database consists of symbols from architectural and electronic drawings represented as graphs. Ending points are represented as nodes and lines and arcs are the edges connecting these ending points. It is composed of 22 classes.

### 3.2 Experimental Protocol

Considering that the complexity and computational time to calculate the distance vectors for the SVM method is soaring, we decide to perform preliminary experiments where we randomly selected two graphs of each class from the training set to be our training, and for our test, we selected 10% of the testing graphs from each class. As we are selecting randomly the training and testing sets, we



need to perform more experiments to obtain an average result, to avoid any bias a unique experiment selecting training and testing sets can have. Thus, we performed each experiments 5 times to obtain our results. To evaluate our approach, we present the mean accuracy score and the standard deviation of a  $k$ -NN classifier ( $k = 3$ ). Table 2 presents detailed information about the datasets.

**Table 2.** Informations about the datasets.

	Datasets				
	Letter-LOW	Letter-MED	Letter-HIGH	Mutagenicity	GREC
# graphs	750	750	750	1500	286
# classes	15	15	15	2	22
# graphs per class	50	50	50	830/670	13
# graphs in learning	30	30	30	4	44
# distance vectors	$\approx 10,000$	$\approx 10,000$	$\approx 10,000$	$\approx 14,000$	$\approx 130,000$
# graphs in testing	75	75	75	129/104	44

### 3.3 Results

In our first experiments, to provide a baseline, we performed the graph matching using the HEOM distance function between the node signatures to populate the cost matrix. We also populated the cost matrix with random values between 0 and 1 for comparison. Table 3 shows these results for the chosen datasets. The HEOM distance approach shows improvement over a simple random selection of values.

**Table 3.** Accuracy results for HEOM distance and random population of the cost matrix in the graph matching problem (in %).

Approach	Datasets				
	Letter-LOW	Letter-MED	Letter-HIGH	Mutagenicity	GREC
Random	$0.53 \pm 0.73$	$1.60 \pm 2.19$	$1.60 \pm 1.12$	$54.85 \pm 4.22$	$1.36 \pm 2.03$
HEOM distance	$40.53 \pm 11.72$	$15.73 \pm 3.70$	$10.93 \pm 3.70$	$49.44 \pm 10.69$	$52.27 \pm 7.19$

As we can see in Table 3, the HEOM distance presents a better result than the random assignment of weights, except for the Mutagenicity dataset, which

is the only dataset with two classes. In this case, the obtained results are similar, considering the standard deviation of the executions ( $\pm 4.22$  for Random approach, and  $\pm 10.69$  for the HEOM approach).

Next, we run experiments using the proposed multi-class SVM approach to compare with the results obtained using the HEOM distance in the cost matrix. We used default parameters for the SVM for the training step (RBF kernel,  $C = 0$ ). We also present results of experiments in which we normalize the distance vector, using min-max (normalizing between 0 and 1) and zscore (normalization using the mean and standard deviation) normalizations. Table 4 shows the mean accuracy of the experiments made.

**Table 4.** Mean accuracy (in %) for the HEOM distance and SVM multi-class approach in the graph matching problem. The best results for each dataset are show in bold.

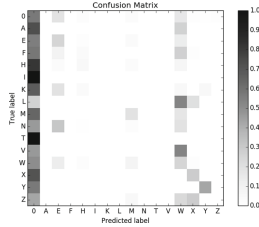
		Datasets				
		Letter-LOW	Letter-MED	Letter-HIGH	Mutagenicity	GREC
HEOM distance		<b>40.53 ± 11.72</b>	15.73 ± 3.70	10.93 ± 3.70	49.44 ± 10.69	<b>52.27 ± 7.19</b>
SVM multi-class	min-max	30.67 ± 5.50	<b>28.00 ± 9.80</b>	18.93 ± 5.77	<b>71.24 ± 29.50</b>	18.64 ± 6.89
		33.33 ± 7.12	20.27 ± 6.69	14.40 ± 5.02	63.26 ± 15.61	20.00 ± 7.43
	zscore	37.87 ± 9.83	21.87 ± 1.52	<b>20.27 ± 8.56</b>	64.12 ± 7.68	30.91 ± 2.59

Table 4 shows us that the SVM approach is promising, obtaining better results for three of the five datasets considered. The improvement in the Mutagenicity dataset was above 20 % points from the HEOM distance baseline. As for the other cases, the Letter-LOW dataset had similar results for the HEOM distance and SVM approach (standard deviation of the HEOM is  $\pm 11.72$  and for the SVM is  $\pm 9.83$ ). The GREC dataset was the only dataset with a distant results from the HEOM approach. We discuss that it is because the dataset has more classes than the others, so its “different” class contains more distance vectors combining node signatures of different classes. With this imbalanced distribution, the “different” class shadows the other classes in the SVM classification.

Table 4 also shows that a normalization step can help separate the classes in the SVM, being successful in improving the result of three of five approaches used, specially the zscore normalization, that considers the mean and standard deviation of the vectors.

To better understand our results, we also calculated the accuracy of the SVM classification for the same training used in it. Our experiments shows that the “different” class does not help the learning, especially in the datasets with more classes, as this “different” class overlook the other classes, preventing the classification as the correct class. It also shows the necessity of a bigger training and a validation set to tune the parameters of the SVM. Figure 2 shows a confusion matrix of a classification of the training data in the Letter-LOW dataset.

To improve our results, we propose to ignore the “different” class in the training set. Table 5 shows the accuracy for this new proposal.



**Fig. 2.** Classification of the training set for the Letter LOW dataset.

**Table 5.** Accuracy scores for four datasets (in %).

Modification	Multi-class	Datasets			
		Letter-LOW	Letter-MED	Letter-HIGH	GREC
Without “different” class	min-max	$37.87 \pm 5.88$	<b><math>34.13 \pm 9.78</math></b>	<b><math>29.07 \pm 4.36</math></b>	$38.18 \pm 8.86$
		$30.13 \pm 6.34$	$30.13 \pm 9.31$	$27.47 \pm 7.92$	$35.45 \pm 2.03$
	zscore	<b><math>44.80 \pm 5.94</math></b>	$25.87 \pm 0.73$	$29.07 \pm 5.99$	$41.82 \pm 7.11$

As we can see in Table 5, our proposed modifications improved the results obtained in our experimental protocol. The dataset Letter-LOW achieved the best result when we do not consider the “different” class in the training step, avoiding misclassification as “different” class. With this, we show that our proposed approach to learn the cost to match nodes are very promising.

## 4 Conclusions

In this paper, we presented an original approach to learn the costs to match nodes belonging to different graphs. These costs are later used to compute a dissimilarity measurement between graphs. The proposed learning scheme combines a node-signature-based distance vector and an SVM classifier to produce a cost matrix, based on which the Hungarian algorithm computes graph similarities. Performed experiments considered the graph classification problem, using k-NN classifiers built based on graph similarities. Promising results were observed for widely used graph datasets. These results suggest that our approach can also be extended to use similar methods based on local vectorial embeddings and can be exploited to compute probabilities as estimators of matching costs.

For future work, we want to perform experiments considering all training and testing sets to compare with our results presented in this paper, and also make a complete study on the minimum training set necessary to achieve a good performance not only in classification, but also in retrieval tasks.

**Acknowledgments.** Experiments presented in this paper were carried out using the Grid’5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER, and several Universities, as well as other organizations (see <https://www.grid5000.fr>).

## References

1. Brun, L., Gaüzère, B., Fourey, S.: Relationships between Graph Edit Distance and Maximal Common Unlabeled Subgraph. Technical report, July 2012
2. Bunke, H.: On a relation between graph edit distance and maximum common subgraph. *Pattern Recogn. Lett.* **18**(8), 689–694 (1997)
3. Bunke, H., Allermann, G.: Inexact graph matching for structural pattern recognition. *Pattern Recogn. Lett.* **1**(4), 245–253 (1983)
4. Bunke, H., Günter, S., Jiang, X.: Towards bridging the gap between statistical and structural pattern recognition: two new concepts in graph matching. In: Singh, S., Murshed, N., Kropatsch, W. (eds.) ICAPR 2001. LNCS, vol. 2013, pp. 1–11. Springer, Heidelberg (2001). [https://doi.org/10.1007/3-540-44732-6\\_1](https://doi.org/10.1007/3-540-44732-6_1)
5. Bunke, H., Riesen, K.: Recent advances in graph-based pattern recognition with applications in document analysis. *Pattern Recogn.* **44**(5), 1057–1067 (2011)
6. Caetano, T.S., McAuley, J.J., Cheng, L., Le, Q.V., Smola, A.J.: Learning graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(6), 1048–1058 (2009)
7. Cho, M., Alahari, K., Ponce, J.: Learning graphs to match. In: IEEE International Conference on Computer Vision ICCV 2013, Sydney, Australia, 1–8 December 2013, pp. 25–32 (2013)
8. Cortés, X., Serratos, F.: Learning graph-matching edit-costs based on the optimality of the oracle’s node correspondences. *Pattern Recogn. Lett.* **56**, 22–29 (2015)
9. Cortés, X., Serratos, F.: Learning graph matching substitution weights based on the ground truth node correspondence. *IJPRAI* **30**(2), (2016)
10. Kuhn, H.W., Yaw, B.: The Hungarian method for the assignment problem. *Naval Res. Logist. Quart.* **2**(1–2), 83–97 (1955)
11. Leordeanu, M., Sukthankar, R., Hebert, M.: Unsupervised learning for graph matching. *Int. J. Comput. Vision* **96**(1), 28–45 (2012)
12. Neuhaus, M., Bunke, H.: Self-organizing maps for learning the edit costs in graph matching. *IEEE Trans. Syst. Man Cybern. Part B* **35**(3), 503–514 (2005)
13. Neuhaus, M., Bunke, H.: Automatic learning of cost functions for graph edit distance. *Inf. Sci.* **177**(1), 239–247 (2007)
14. Neuhaus, M., Bunke, H.: Bridging the Gap Between Graph Edit Distance and Kernel Machines. World Scientific Publishing Co., Inc., River Edge (2007)
15. Riesen, K., Bunke, H.: Iam graph database repository for graph based pattern recognition and machine learning. In: da Vitoria Lobo, N., et al. (eds.) SSPR /SPR. LNCS, vol. 5342, pp. 287–297. Springer, Heidelberg (2008). [https://doi.org/10.1007/978-3-540-89689-0\\_33](https://doi.org/10.1007/978-3-540-89689-0_33)
16. Riesen, K., Ferrer, M.: Predicting the correctness of node assignments in bipartite graph matching. *Pattern Recogn. Lett.* **69**, 8–14 (2016)
17. de Sa, J.M.: *Pattern Recognition: Concepts, Methods, and Applications*. Springer Science & Business Media, Berlin (2001). <https://doi.org/10.1007/978-3-642-56651-6>
18. Silva, F.B., de Oliveira Werneck, R., Goldenstein, S., Tabbone, S., da Silva Torres, R.: Graph-based bag-of-words for classification. *Pattern Recogn.* **74**, 266–285 (2018)
19. Wilson, D.R., Martinez, T.R.: Improved heterogeneous distance functions. *J. Artif. Int. Res.* **6**(1), 1–34 (1997)

# **Multimedia Analysis and Understanding**



# Matrix Regression-Based Classification for Face Recognition

Jian-Xun Mi<sup>(✉)</sup>, Quanwei Zhu, and Zhiheng Luo

Chongqing University of Posts and Telecommunications, Chongqing 400065, China  
mijianxun@gmail.com, quanwei12@gmail.com, zhiheng.luo2@gmail.com

**Abstract.** Partially occlusion is a common difficulty arisen in applications of face recognition, and many algorithms based on linear representation may pay attention to such cases. In this paper, we consider the partial occlusion problem via inner-class linear regression. Specifically, we develop a matrix regression-based classification (MRC) method in which every sample from the same class are represented as matrices instead of vector and adopted to encode a probe image under. In the regression step, a L21-norm based matrix regression model is proposed, which can efficiently depress the effect of occlusion in probe image. Accordingly, an efficient algorithm is derived to optimize the proposed objective function. In addition, we argue that the corrupted pixels in probe image should not be considered in decision step. Thus, we introduce a robust threshold to dynamically eliminate the corrupted rows in probe image before making decision. Performance of MRC is evaluated on several datasets and the results are compared with those of other state-of-the-art methods.

## 1 Introduction

Recently, face recognition (FR) has been widely used in many fields [3, 14]. However, robust face recognition is still a difficult problem due to the varied noises, such as real disguise, continuous or pixel-wise occlusion. In such case, it is usually unable to know the occlusion position and the percentage of occluded pixels in advance.

For FR, samples from a specific subject can be assumed to lie in a subspace of all the face space [1, 2]. So, a coming probe image can be well represented as a linear combination of all images from the same class. Based on this assumption, linear representation based FR methods arise. These methods can be categorized into two groups: collaborative representation and inner-class representation. Collaborative representation uses whole gallery images to represent probe image while inner-class representation query image by the linear combination of class-specific images superlatively.

The most typical approach of collaborative representation is the sparse representation classification (SRC) [15]. SRC selects a part of training samples that are strongly competitive to represent a query image. Then the decision is made by identifying which subject yields the minimal reconstruction residual. In SRC,

linear regression uses L1-norm as the regularization term, which is also called Lasso problem. SRC believes that this regularization technique makes the coefficients sparse and sparse coefficients are more discriminative in classifying. However, in the later research, Zhang et al. [18] argue that it is the collaborative representation rather than sparsity that contributes to classifying. They proposed collaborative representation based classification (CRC), which applying L2-norm constraint to representation coefficients and obtaining a competitive result. Compared with SRC which solves an optimization with an iterative algorithm, CRC has a closed-form solution.

Following SRC and CRC, Yang et al. [16] propose nuclear norm based matrix regression (NMR) classification framework by applying nuclear norm on residual errors. NMR shows better FR performance in the presence of occlusion and illumination variations. He et al. [5] proposed Correntropy-Based Sparse Representation (CESR) which combines the maximum correntropy criterion with a nonnegative constraint on representation vector to obtain a sparse representation. Yang et al. [17] propose the Regularized Robust Coding (RRC) which determines the representation coefficient with maximum a posterior (MAP) estimation to get a good fidelity term and use a flexible shape to describe the distribution of residual error.

Apart from collaborative representation methods, inner-class representation methods such as linear regression based classification (LRC) [8] also have good performance in FR. Unlike collaborative representation methods, in LRC probe images are represented by a special class at each time. Although collaborative representation makes all training samples compete with each other, which is beneficial to produce a discriminative representation vector, a drawback is that once dealing with an occluded probe the representation residual contains both within-class variation and between-class variation. Besides, at representation step, the produced coding coefficient vector is not aware any information of class label. That is to say, the permutation of training samples is ignored at representation step. Those drawbacks may lead to misclassification. For LRC, the representation residual from the correct class contains only within-class variation while those from the other classes contain both within-class variation and between-class variation. Thus, residual error in the correct class should be the smallest one and that is helpful for classification.

Most of the mentioned methods treat images as vectors which ignores the existent correlation among pixels. Occlusions such as sunglasses, scarf and veil are always structural. So, we argue that classifier should preserve the two-dimensional (2D) correlation. On the other hand, in those approaches, all the pixels on the probe sample are used to classify probe samples. In the case where probe samples with occlusion, it is hard to guarantee the stability of these methods since occlusion part could unpredictably favor some classes. So, we introduce dynamic threshold to ensure occlusion is entirely depressed. Combining the two points, we develop a novel method named Matrix-based Linear Regression (MRC) which treats all image as matrices. In representation step, a probe image is regressed as a linear combination of samples from each class and MRC

uses L21-norm to compute the regression loss. Finally, dynamically threshold is employed to eliminate occlusion before decision step. Three main contributions of MRC are outlined as follows:

- (1) MRC represents every image as a 2-D matrix. Pixels in a local area of an occlusion image are generally highly correlated. Transforming the image as a vector may discard those correlations while 2-D matrix can preserve does not.
- (2) MRC uses L21-norm based regression loss. L21-norm has two advantages: the robust nature of L1-norm, which is efficient for error detection, and the ability of preserving the spatial information. The use of L21-norm based regression loss can depress the effect of occlusion in regression step.
- (3) MRC employs a self-adaptive threshold to construct a robust classifier. As we claim, corrupted pixel should not participate in classifying. The threshold restricts large residual error dynamically before our decision step. In this way, MRC can be more robust to occlusion.

The rest of paper is organized as follows: In Sect. 2, we review some related works. In Sect. 3, we present the MRC model with an effective solution. In Sect. 4, we conduct extensive experiments. Finally, the conclusion is drawn in Sect. 5.

## 2 Related Work

### 2.1 L21-Norm

L21-norm is an element-wise matrix norm and has been used in feature selection and other machine learning topics for years [9,11]. For a matrix  $M \in R^{m \times n}$ , the norm can be defined as  $\|M\|_{2,1} = \sum_{i=1}^m \sqrt{\sum_{j=1}^n \|M_{i,j}\|_2}$ , where  $M_{i,j}$  donates elements located in the  $i$ -th row and the  $j$ -th column. L21-norm can be seen as a balance between L1-norm and L2-norm.

### 2.2 LRC

LRC is an inner-class linear regression model. Assume there are  $N$  number of distinguished classes with  $p_i$  number of training images from the  $i$ -th class. Each training image is transform into a  $m$ -dimensional vector so the  $i$ -th class samples can be described as  $X_i = [x_1, x_2, \dots, x_{p_i}] \in R^{m \times p_i}$ , where  $x_{p_i}$  is the  $p_i$ -th image in the class. Given a probe image  $y \in R^{m \times 1}$ , LRC regresses  $y$  with training images from each class:  $y = X_i \beta_i$ , where  $\beta_i$  is the coefficient of  $y$  in  $i$ -th class. LRC uses  $\beta_i$  to predict the response vector for each class as  $\hat{y}_i = X_i \beta_i$ . Then LRC calculates the distance between the predicted response vector  $\hat{y}_i$  and the original response vector  $y$ :

$$d_i(y) = \|y - \hat{y}_i\|_2, \quad i = 1, 2, \dots, N \quad (1)$$

Finally, the class label of  $y$  is determined by the class with minimum distance:

$$ID(i) = \min d_i(y) \quad (2)$$



### 3 Matrix-Based Linear Regression

In this section, we first present the motivation of MRC. Then, we give the objective function of our model. Finally, an iterative optimal solution is given for MRC.

#### 3.1 Motivation of MRC

As the previous statement, linear representation is easily affected by serious occlusion, in order to decrease the influence, we introduce L21-norm to inner-class representation and treat images as matrices. Real disguise can be approximately considered as some row occlusion in an image. If we consider an image as a matrix, regression under L21-norm constraint can easily depress the influence of row occlusion. Another problem is that the residuals corresponding to corrupted parts will be very large and make classification difficult. We argue that large residuals should not be taken into consideration during decision step. Therefore, a robust threshold is employed to restrict the large residuals.

#### 3.2 Proposed MRC

Follow the previous thoughts, we now develop the MRC model. First, we introduce some denotations. Assume the training set contains images belonging to  $N$  classes and each class including  $p_i$  images. The image size is  $m \times n$ .  $A^{i,j} \in R^{m \times n}$  represents the  $j$ -th image in the  $i$ -th class. For computing convenience, we define matrix  $D_l^i \in R^{p_i \times n}$  which is the combine of the  $l$ -th row in the  $i$ -th class. More specifically, we stack all images in the  $i$ -th class and extract the  $l$ -th row of all images to construct  $D_l^i$  (see Fig. 1). Given a probe image  $Y \in R^{m \times n}$ ,  $Y$  is regressed in each class as follows:

$$\min \|Y - \sum_{j=1}^{p_i} A^{i,j} x^{i,j}\|_{2,1}, \quad i = 1, 2, \dots, N \quad (3)$$

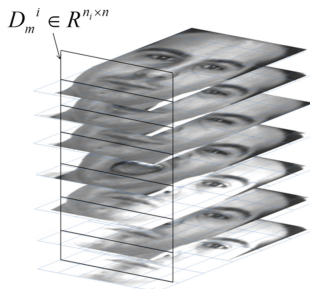


Fig. 1. An illustration of  $D_l^i$ .

where  $x^{i,j}$  is the corresponding coefficient of  $A^{i,j}$ . Equation (3) can be reformulated as

$$\min \sum_{l=1}^m \|Y_l - X_i^T D_l^i\|_2, \quad i = 1, 2, \dots, N \tag{4}$$

where  $Y_l$  is the  $l$ -th row of  $Y$  and  $X_i = [x^{i,1}, x^{i,2}, \dots, x^{i,p_i}]^T$ . Then we propose an iterative reweight method to solve Eq. (4). We introduce an auxiliary variable

$$w_l^i = \frac{1}{\|Y_l - X_i^T D_l^i\|_2} \tag{5}$$

and Eq. (4) becomes

$$\min \sum_{l=1}^m \|Y_l - X_i^T D_l^i\|_2 = \min \sum_{l=1}^m w_l^i \|Y_l - X_i^T D_l^i\|_2^2 \tag{6}$$

We first fix  $w_l^i$  and minimize Eq. (6) to obtain the  $X_i$ . Now we take derivative of Eq. (6) with the respect to  $X_i$  and set it to zeros. Then, we get

$$X_i = \left( \sum_{l=1}^m w_l^i D_l^i (D_l^i)^T \right)^{-1} \left( \sum_{l=1}^m w_l^i D_l^i y_l^T \right) \tag{7}$$

After computing  $X_i$  we go back to update  $w_l^i$  according to Eq. (5). Then we repeated update  $X_i$  and  $w_l^i$  until converge. We outline the algorithm in algorithm 1.

---

**Algorithm 1.** Reweighted algorithm for MRC in  $i$ -th class

---

**Input:** Dataset  $D_l^i$ , probe image  $Y$ .

- 1: initial  $X_i$  with a random vector
- 2: **while** not converge **do**
- 3:   calculate  $w_l^i$  according to Eq.(5).
- 4:   calculate  $X_i$  according to Eq.(7).
- 5: **end while**

**Output:** The coefficient of  $i$ -th class:  $\hat{X}_i$

---

Based on  $\hat{X}_i$ , we can make the decision of the label by using the nearest subspace criterion under L21-norm.  $\hat{X}_i$  along with the  $A^{i,j}$  is used to calculate the residual error for each class,

$$e^i = y - \sum_{j=1}^{p_i} A^{i,j} x^{i,j} \tag{8}$$

$$d(i) = \|e^i\|_{2,1} \tag{9}$$

In previous methods using NS decision rules, such as LRC and SRC,  $y$  is assigned to class with minimum  $d(i)$ . However, as we claim before, the residuals

are produced not only by fidelity pixels but also complexity noises. The distances between probe image and its representation could not reflect the real conditions by putting all the residuals into the measurement. In order to ensure make the classification result is stable and reliable, only the representation residuals of the fidelity pixels should be taken into consideration during decision. In MRC, thanks to the L21-norm constraint, residuals corresponding to occlusion parts will be very large, which provides evidence to possibly remove the occlusion.

Here, we let MRC adopt a threshold to crop the large residuals. A natural thought is to set the threshold to mean of residuals. However, the mean of data can be easily affected by extreme. To achieve robust detection of occlusion, we consider a robust estimation of the non-contaminated part of facial feature by setting a threshold under which only small Gaussian noise passes, not the occlusion. Therefore, in MRC, the median absolute deviation (MAD), which also is known as a robust estimation of standard deviation, is employed. MAD can be used to detect outliers [6]. Given data  $a$ , its MAD is calculated as:

$$mad(a) = median(|a - median(a)|) \quad (10)$$

where  $median(\cdot)$  aims to find median value of the data.

Now we put MAD into MRC. Equation (9) can be seen as a two step procedure. First, calculate L2-norm of each row of  $e^i$  then sum up all the results. The L2-norm of the occlusion rows would be large than other rows. Then we apply MAD threshold to the L2-norm of each row before summing them up. The Eq. (9) becomes

$$\xi_l^i = \|e_l^i\|_2 \quad (11)$$

where  $\xi_l^i$  is the  $l$ -th row of  $\xi^i$ . We define the threshold on as

$$threshold = median(\xi^i) + k \times mad(\xi^i) \quad (12)$$

where  $k \in [0, 1]$  is a parameter to adjust the ratio between the two statistics. And we apply threshold to  $\xi_l^i$ :

$$\xi_l^i = \begin{cases} \xi_l^i, & \xi_l^i < threshold \\ 0, & \xi_l^i > threshold \end{cases} \quad (13)$$

$$\hat{d}(i) = \|\xi^i\|_1 \quad (14)$$

Finally MRC assigns  $y$  to the class with minimum  $\hat{d}(i)$

$$label = \arg \min(\hat{d}(i)) \quad (15)$$

Here we outline the MRC classification algorithm in Algorithm 2.

## 4 Experiments

In this section, we perform experiments on face databases to demonstrate the performance of MRC. We first evaluate MRC for FR under different sizes of

---

**Algorithm 2.** MRC Classification algorithm

---

**Input:** Dataset  $A$ , probe image  $Y$ .

- 1: **for all** each class in  $A$  **do**
- 2:   Construct  $D_i^l$ .
- 3:   Compute  $\hat{X}_i$  according to algorithm 1.
- 4:   Compute  $\xi^i$  according to Eq.(8) and Eq.(11).
- 5:   Compute threshold of  $\xi^i$  according to Eq.(12).
- 6:   Cope  $\xi^i$  according Eq.(13)
- 7:   Compute distance  $\hat{d}(i)$  according to Eq.(14).
- 8: **end for**
- 9: Categorize  $Y$  according to Eq.(15)

**Output:** Class of  $Y$

---

simulated occlusion. Further, we carry out experiments under real disguise to demonstrate the robustness of MRC. The proposed MRC is compared to related existing methods including SRC [15], CRC [18], LRC [8], RRC [17], and CESR [5]. Five standard databases, including the AR face [7], The CMU PIE face [13], the Extended Yale B database [4] the ORL database [12] and the FERET database [10] are employed to evaluate the performance of these methods.

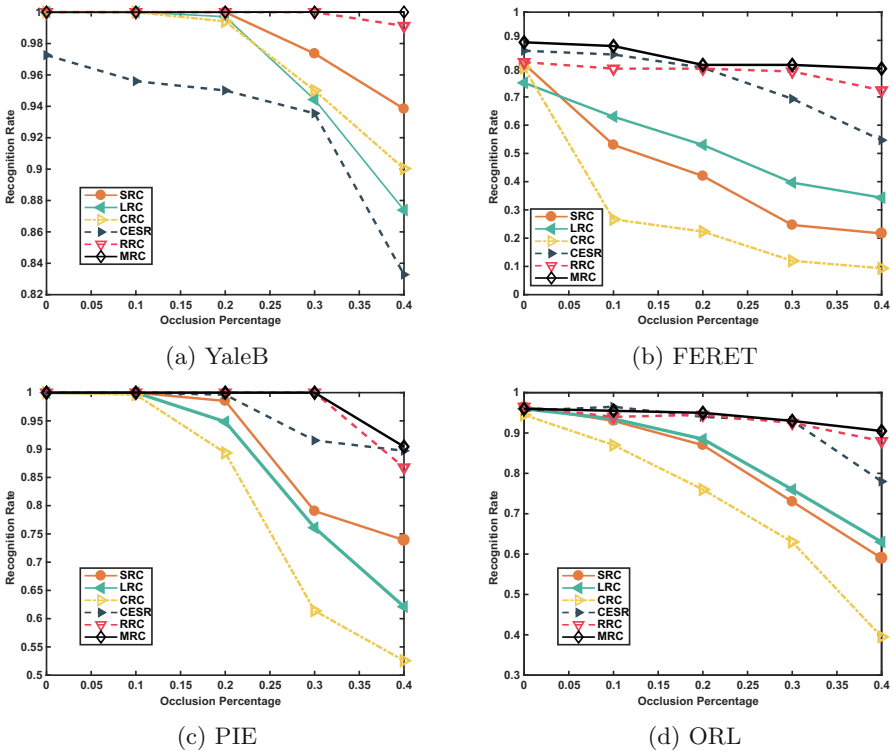
#### 4.1 Recognition with Row Occlusions

We carry out the first experiment in FR with row occlusions. The YaleB database, the PIE database, the ORL database and the FERET face database are employed for this purpose. In the first experiments, for each probe image, we randomly set a certain percentage of its row to zeros. We run the experiments 10 times and the average recognition rates are shown in Fig. 2 It can be seen that MRC achieve the highest recognition rates among all methods in all dataset. When the occlusion rate is zero all methods perform well. But with increasing of occlusion, the recognition rate of SRC, CRC and LRC decreases sharply. The CESR method shows its robustness to occlusion in FERET, PIE and ORL dataset. The RRC method has the almost same performance as MRC. However, MRC has an improvement of it over with respectively 0.009%, 0.07%, 0.04%, 0.03% in the four datasets.

#### 4.2 Recognition with Block Occlusions

From the first experiments results, we can see that MRC has strong robustness to deal with large-scale line-based occlusions. In the second experiments validate the robustness of MRC to block occlusions. In this experiment, we choose subset 1 of Yale dataset as the training set. And subset 2 and subset 3 with various sizes of block are selected as test set respectively. We vary the block size from 10% to 40% of an image. The experiment is run 10 times and the average results are shown in Table 1.

Subset 1, 2, 3 of YaleB are with few illumination changes. So it is easy to obtain high recognition rate in the subsets. We can observe for the table that



**Fig. 2.** Face recognition rate versus with the row occlusion percentage ranging from 10% to 40% in Yale, FERET, PIE and ORL.

**Table 1.** Recognition rate with block occlusions.

Methods	Subset2				Subset3			
	10%	20%	30%	40%	10%	20%	30%	40%
LRC	81.72	79.301	77.957	72.043	77.688	75.269	70.699	68.28
CRC	71.237	72.312	69.624	53.226	58.602	55.108	50.538	34.409
RRC	<b>100</b>	<b>100</b>	<b>100</b>	99.462	<b>100</b>	99.731	99.194	95.161
CESR	99.731	98.656	97.849	97.043	68.548	63.978	64.247	55.914
SRC	76.344	70.968	67.473	56.183	62.366	58.602	56.72	54.57
MRC	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>98.387</b>

MRC achieve 100% recognition rate except for one case. Similar to the first experiment, MRC outperforms all other methods. SRC, LRC and CRC are not good at resisting the block occlusion. In subset 2, the CESR method has high recognition rate when 40% of an image is occupied. While in subset 3 the CESR only obtain 55.91% recognition rate under the same condition. The RRC method

also has good performance with less occlusion. In subset 3, it is equal to MRC when the occlusion percent is 10%. When the occlusion percent is 20%, 30% and 40%, MRC has an improvement of 0.27%, 0.81% and 4.84% over RRC, respectively.

### 4.3 Recognition with Real Disguises

After experimenting with random row occlusion and block occlusion scenarios, we further test different approaches in coping with real possible disguise. In this experiments, AR dataset is employed. The dataset contains samples wearing scarf and glasses. We choose images which do not have any occlusion from each subject for training and 6 images were scarf or glasses from each subject for validation. The scale of occlusion by sunglasses and scarf about 20% and 40% respectively. The average recognition rates of 10 runs are shown in Table 2.

**Table 2.** Recognition rate in AR

Method	SRC	LRC	CRC	CESR	RRC	MRC
Recognition rate (%)	50.75	38	74.75	60.75	95.5	<b>96.25</b>

The difficulty in AR dataset not only because probe images contain glass and scarf but there are illumination and expression changes. This may make classifiers misclassification. Taking into account such a complex situation, all the used methods faced a huge challenge. The performances of some algorithms are not satisfactory. However, MRC has an advantage over all methods in this experiment. The proposed MRC approach copes well with the real disguise, achieving high recognition rates of 96.25%, which is 40%, 58%, 22%, 36% and 1% higher than SRC, LRC, CRC, CESR and RRC, respectively. The high recognition rate of MRC indicates the proposed method are robust to real disguises.

## 5 Conclusion

In this paper, we propose a novel classification-based method (MRC) for face recognition which considers classifying probe images as a problem of matrix-based linear regression. The MRC algorithm is extensively evaluated using the standard five databases and compared with the state-of-the-art methods. The experimental results prove our viewpoint that the structural information is useful for face recognition. The good performance of MRC benefits from the combination of the matrix representation and L21-norm fidelity term, which can detect errors and make sure the face features are represented in the matrix regression. The dynamic selection of the representation residuals by the self-adaptive classifier also provides more discriminative information.

## References

1. Basri, R., Jacobs, D.W.: Lambertian reflectance and linear subspaces. *IEEE Trans. Pattern Anal. Mach. Intell.* **25**(2), 218–233 (2003)
2. Belhumeur, P.N., Hespanha, J.P., Kriegman, D.J.: Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(7), 711–720 (1997)
3. De La Torre, F., Black, M.J.: A framework for robust subspace learning. *Int. J. Comput. Vis.* **54**(1–3), 117–142 (2003)
4. Georghiades, A.S., Belhumeur, P.N., Kriegman, D.J.: From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell.* **23**(6), 643–660 (2001)
5. He, R., Zheng, W.S., Hu, B.G.: Maximum correntropy criterion for robust face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(8), 1561–1576 (2011)
6. Leys, C., Ley, C., Klein, O., Bernard, P., Licata, L.: Detecting outliers: do not use standard deviation around the mean, use absolute deviation around the median. *J. Exp. Soc. Psychol.* **49**(4), 764–766 (2013)
7. Martinez, A.M.: The AR face database. CVC Technical report (1998)
8. Naseem, I., Togneri, R., Bennamoun, M.: Linear regression for face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(11), 2106–2112 (2010)
9. Nie, F., Huang, H., Cai, X., Ding, C.H.: Efficient and robust feature selection via joint L<sub>2</sub>, 1-norms minimization. In: *Advances in Neural Information Processing Systems*, pp. 1813–1821 (2010)
10. Phillips, P.J., Wechsler, H., Huang, J., Rauss, P.J.: The feret database and evaluation procedure for face-recognition algorithms. *Image Vis. Comput.* **16**(5), 295–306 (1998)
11. Ren, C.X., Dai, D.Q., Yan, H.: Robust classification using L<sub>2</sub>, 1-norm based regression model. *Pattern Recogn.* **45**(7), 2708–2718 (2012)
12. Samaria, F.S., Harter, A.C.: Parameterisation of a stochastic model for human face identification. In: *Applications of Computer Vision Proceedings of the Second IEEE Workshop on 1994*, pp. 138–142. IEEE (1994)
13. Sim, T., Baker, S., Bsat, M.: The CMU pose, illumination, and expression (PIE) database. In: *Proceedings Automatic Face and Gesture Recognition Fifth IEEE International Conference on 2002*, pp. 53–58. IEEE (2002)
14. Turk, M., Pentland, A.: Eigenfaces for recognition. *J. Cogn. Neurosci.* **3**(1), 71–86 (1991)
15. Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y.: Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(2), 210–227 (2009)
16. Yang, J., Luo, L., Qian, J., Tai, Y., Zhang, F., Xu, Y.: Nuclear norm based matrix regression with applications to face recognition with occlusion and illumination changes. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(1), 156–171 (2017)
17. Yang, M., Zhang, L., Yang, J., Zhang, D.: Regularized robust coding for face recognition. *IEEE Trans. Image Process.* **22**(5), 1753–1766 (2013)
18. Zhang, L., Yang, M., Feng, X.: Sparse representation or collaborative representation: which helps face recognition? In: *IEEE international conference on 2011 Computer vision (ICCV)*, pp. 471–478 IEEE (2011)



# Plenoptic Imaging for Seeing Through Turbulence

Richard C. Wilson<sup>(✉)</sup> and Edwin R. Hancock

University of York, York, UK  
richard.wilson@york.ac.uk

**Abstract.** Atmospheric distortion is one of the main barriers to imaging over long distances. Changes in the local refractive index perturb light rays as they pass through, causing distortion in the images captured in a camera. This problem can be overcome to some extent by using a plenoptic imaging system (one which contains an array of microlenses in the optical path). In this paper, we propose a model of image distortion in the microlens images and propose a computational method for correcting the distortion. This algorithm estimates the distortion field in the microlenses. We then propose a second algorithm to infer a consistent final image from the multiple images of each pixel in the microlens array. These algorithms detect the distortion caused by changes in atmospheric refractive index and allow the reconstruction of a stable image even under turbulent imaging conditions. Finally we present some reconstruction results and examine whether there is any increase in performance from the camera system. We demonstrate that the system can detect and track distortions caused by turbulence and reconstruct an improved final image.

## 1 Introduction and Related Work

It is an unfortunate fact for long-range high magnification imaging that the atmosphere perturbs light as it passes through. This is well known to astronomers, who go to great lengths to find locations with optimum viewing conditions. When light passes through the atmosphere, it is bent by areas of different refractive indices caused by pressure differences. Long-range imaging with normal cameras suffers greatly from atmospheric distortion, as the distance which the light rays travel through the atmosphere is generally long. This is particularly apparent, for example, when the ground is warmed by the sun and causes turbulent convection [1].

A number of solutions have been proposed to this problem. Lucky imaging [6] relies on identifying short windows of time when the conditions are optimal and sharp images can be recovered. The turbulence is chaotic and there are moments when the distortion subsides and a clear image can be captured. This, however, limits the rate at which data can be captured. Another approach is speckle interferometry aims to reconstruct an image from multiple short exposures [7]. This is based on the fact that the largest atmospheric distortions are at low



frequencies. The high frequency information present in the images is combined to form one high resolution image.

The modern solution is to use adaptive optics. In an adaptive system, the shape of the reflector can be rapidly altered to compensate for the wavefront distortion introduced by the atmosphere. This results in a sharp image at the sensing plane. The shape of the wavefront is determined by using a wavefront sensor (for example a Shack-Hartmann device [2]). This device uses a multi-lens array and light sensor to detect the local slope of the wavefront at various positions across the aperture. Essentially it is a plenoptic camera. Although the plenoptic camera is a very old concept, it has risen in popularity over the last two decades as the computational power has become available to process the plenoptic images [3,4]. A plenoptic or light-field camera is a camera which is capable of capturing more than the usual 2D image of a scene. The plenoptic camera can determine both the intensity of light in the image and the direction with which rays strike the image. This is usually achieved using an array of microlens behind the main objective lens; the microlenses separate out different ray directions before they strike the image plane.

An alternative to these mechanical systems is to use computational imaging. Statistical methods can be used in place of expensive hardware to reconstruct the images captured by a plenoptic camera. Previous work in this area [8,9] has used a plenoptic camera to reduce the distortion captured in the image plane. Lucky imaging is then used to locate pixels from individual cells which are well imaged. This overcomes some of the problems with waiting time for lucky imaging.

The goal of this work is to propose a statistical model of the images captured by plenoptic cameras and use this model to predict and reconstruct undistorted images from the data. In Sect. 3, we develop a model of the microlens images which exploits a Gaussian process model and the sparsity of the problem to find the distortion present in each micro-image. In Sect. 4, we propose a linear model to reconcile the final image with the multiple microlens images and their distortion models. In Sect. 5, we present reconstruction results on experimental data.

## 2 Microlens Image Matching

### 2.1 Image Formation

The action of a plenoptic or lightfield camera can be described by an analysis of the lightfield as it passes through the camera [5,10]. The lightfield describes the position and direction of light rays as they pass through a particular plane of the imaging system. We can describe the lightfield which enters the camera at the objective as  $r(\mathbf{q}, \mathbf{p})$  which gives the intensity of the ray at position  $\mathbf{q}$  travelling in direction  $\mathbf{p}$ . After travelling through the optical system, the lightfield at the sensor  $s$  is

$$r_s(\mathbf{q}, \mathbf{p}) = r\left(-\frac{a}{b}\mathbf{q}, \frac{1}{f}\mathbf{q} - \frac{b}{a}\mathbf{p}\right). \quad (1)$$

Here  $a$  and  $b$  are the distances from the primary focus to the microlens and microlens to image plane respectively, and  $f$  is the microlens focal length.

Since the sensor is not sensitive to direction, we obtain the sensed intensity at position  $\mathbf{b}$  by integrating over all directions  $\mathbf{p}$  incident at  $\mathbf{q}$ , to give

$$I_s(\mathbf{q}) = \frac{d}{b} \bar{r}\left(\frac{a}{b}\mathbf{q}, \frac{1}{b}\mathbf{q}\right), \quad (2)$$

where  $\bar{r}(\cdot)$  indicates the intensity function averaged over all directions incident at that point and  $d$  is the microlens diameter. As a result, by sampling at different positions  $\mathbf{q}$  we can obtain information about both ray position and direction, each sampled at a rate determined by  $a$  and  $b$ . Atmospheric distortion causes two effects in these images. Firstly there is an overall shift in the position of each microlens image due to the (distorted) angle of the incoming wavefront. Secondly, there is local distortion caused by the small scale variations in the phase over the microlens. Our goal is therefore to detect the overall shift of the microlens image in a way that is robust to local distortions.

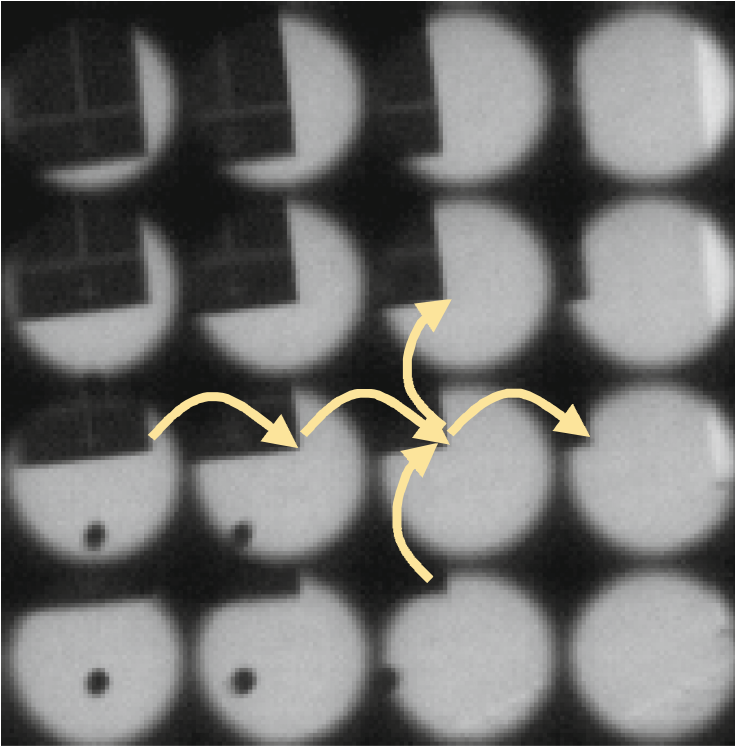
## 2.2 Distortion Model

We begin by finding the correspondence between pairs of microlens images (the source and the target), in order to find the relative shift of pixels between the pair. The shift is estimated in two parts; the overall shift of the microlens image is  $\mathbf{s} = (s_x, s_y)^T$ . The shift of an individual pixel  $i$  within the microlens image is given by  $(x_i, y_i)^T$ . The local distortion at  $i$  is then given by  $(x_i, y_i)^T - \mathbf{s}$ . The pixel shifts are encoded in an interleaved long-vector

$$\mathbf{x} = \begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \end{pmatrix}. \quad (3)$$

In order to estimate these pixel shifts, we need to match points between neighbouring microlens images. This is illustrated in Fig. 1. A local residual between point  $i$  in the first microlens image and the second image is found using local 5 by 5 block matching:

$$\begin{aligned} R(\Delta x, \Delta y) &= \sum_{k,l=-2\dots 2} [I(x_i + o_x + k + \Delta x, y_i + o_y + l + \Delta y) \\ &\quad - I(x_i + k, y_i + l)]^2, \end{aligned} \quad (4)$$



**Fig. 1.** A portion of the plenoptic image, showing a 4 by 4 array of microlens images and the match between points in neighbouring microlenses. The matching point corresponds to the upper left door corner in Fig. 2.

where  $(o_x, o_y)$  is the offset from the source microlens image to the target. The residuals  $R(\Delta x, \Delta y)$  are assumed to follow a 2D Normal distribution and from this distribution we find a mean offset  $\boldsymbol{\mu}_i$  and variance  $\boldsymbol{\Sigma}_i$  of the matching position for each pixel. Smoothness is imposed on the field of local distortions using a Gaussian prior:

$$C(x, y; a, b) = \exp \left[ -\frac{(x - a)^2 + (y - b)^2}{2\sigma^2} \right]. \tag{5}$$

Putting these ingredients together, we have a Gaussian process log-likelihood for the shift and distortion of

$$L = (\mathbf{x} - s_x \mathbf{1}_X - s_y \mathbf{1}_Y)^T \mathbf{C}^{-1} (\mathbf{x} - \alpha \mathbf{1}_X - \beta \mathbf{1}_Y) + (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}), \tag{6}$$

where

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \\ \vdots \end{pmatrix}, \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_1 & 0 & 0 \\ 0 & \boldsymbol{\Sigma}_2 & \\ 0 & & \ddots \end{pmatrix},$$

$$\mathbf{1}_X = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix}, \mathbf{1}_Y = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ \vdots \end{pmatrix}.$$

The first part of the log-likelihood enforces smoothness on the recovered shift vector  $\mathbf{x}$ , and the second part ensures that the shifts match similar areas of the microlens images. Maximum-likelihood estimation is relatively straightforward and gives the following equations for  $\mathbf{s}$  and  $\mathbf{x}$ :

$$(\mathbf{C}^{-1} + \boldsymbol{\Sigma}^{-1} - T) \mathbf{x} = \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} \quad (7)$$

$$\mathbf{s} = \mathbf{S}^{-1} \begin{pmatrix} \mathbf{1}_X^T \\ \mathbf{1}_Y^T \end{pmatrix} \mathbf{C}^{-1} \mathbf{x} \quad (8)$$

with

$$\mathbf{S} = \begin{pmatrix} \mathbf{1}_X^T \mathbf{C}^{-1} \mathbf{1}_X & \mathbf{1}_X^T \mathbf{C}^{-1} \mathbf{1}_Y \\ \mathbf{1}_Y^T \mathbf{C}^{-1} \mathbf{1}_X & \mathbf{1}_Y^T \mathbf{C}^{-1} \mathbf{1}_Y \end{pmatrix}$$

$$\mathbf{T} = \mathbf{C}^{-1} (\mathbf{1}_X \mathbf{1}_Y) \mathbf{S}^{-1} (\mathbf{1}_X \mathbf{1}_Y)^T$$

This is a large linear system and is expensive to compute. However,  $\mathbf{C}^{-1}$  can be pre-computed and sparsified by dropping small values. As the smoothing range is not normally that large, typically  $\mathbf{C}^{-1}$  can be made quite sparse without affecting the accuracy of the computation.  $\boldsymbol{\Sigma}$  is naturally sparse. As a result, Eq. 7 is the solution of a sparse system of equations which is solved efficiently using a sparse LU decomposition. This is important because of the high frame rate produced by the camera and the consequently large amounts of data produced.

### 3 Image Reconstruction

The result of the above calculations is a set of predicted correspondences between the pixels in pairs of microlens images. In order to reconstruct the final image, we need to map each pixel onto its location in the final image. This means constructing a mapping for each microlens image which respects the pairwise correspondence between images. However, each final position corresponds to pixels in multiple microlens images and the pairwise correspondences may not all be completely consistent due to distortion and the mis-identification of matches.

In a standard plenoptic reconstruction, each microlens pixel has a fixed position in the reconstructed image, determined by the optical parameters and the distance of the imaged object. Two neighbouring microlens images partially overlap with an offset determined by the geometry of the microlens and the parameters  $a$  and  $b$ . We denote this standard position by

$$\mathbf{z}^{(0)} = \begin{pmatrix} \mathbf{z}_1^{(0)} \\ \mathbf{z}_2^{(0)} \\ \vdots \end{pmatrix}, \quad (9)$$

where  $\mathbf{z}_i^{(0)}$  is the usual position (in the reconstructed image) of pixel  $i$  from the microlens array.

In order to determine the positions of pixels in our shifted and distorted microlens images, we need to additionally account for the recovered distortion  $\mathbf{x}$ . Our recovered pixel positions are given by  $\mathbf{z}$ ; i.e.  $z_i$  is the location of pixel  $i$  in the recovered image.

The first step is to use the distortion map  $\mathbf{x}$  to infer a set of correspondences between pairs of pixels in two microlens images. Using these correspondences we construct a matching matrix  $\mathbf{M}$  with entries

$$M_{ij} = \begin{cases} 1 & \text{if } i \text{ matches to } j \\ 0 & \text{otherwise.} \end{cases} \quad (10)$$

If all correspondences are consistent, then matching pixels will be placed in the same location and  $z_i = z_j$  whenever  $M_{ij} = 1$ . Because of inconsistent matches caused by mis-matches and distortion, in practice it is not possible to set all matching pairs equal. Instead we try to minimise the squared difference  $\sum_{ij} M_{ij} (z_i - z_j)^2$ .

This criterion enforces similarity of position for corresponding pixels, but does not determine the overall layout of the pixels in the final image. We therefore look for a solution for  $\mathbf{z}$  that is close to  $\mathbf{z}_0$  so as to preserve the original layout of the image as much as possible. This is essentially a smoothness constraint on the final solution. The optimal solution is found from

$$\mathbf{z}^* = \arg \min_{\mathbf{z}} \left[ \sum_{ij} M_{ij} (z_i - z_j)^2 + \lambda (\mathbf{z} - \mathbf{z}_0)^T (\mathbf{z} - \mathbf{z}_0) \right], \quad (11)$$

which again can be calculated as the solution to a sparse linear system:

$$(\mathbf{D} - \mathbf{M} + \lambda \mathbf{I}) \mathbf{z} = \lambda \mathbf{z}_0, \quad (12)$$

where  $\mathbf{D}$  is the diagonal matrix with  $D_i = \sum_j M_{ij}$ , i.e. the number of matches for pixel  $i$ .

As the last step, a final image is reconstructed by projecting each pixel from the multilens image into the final image and interpolating.

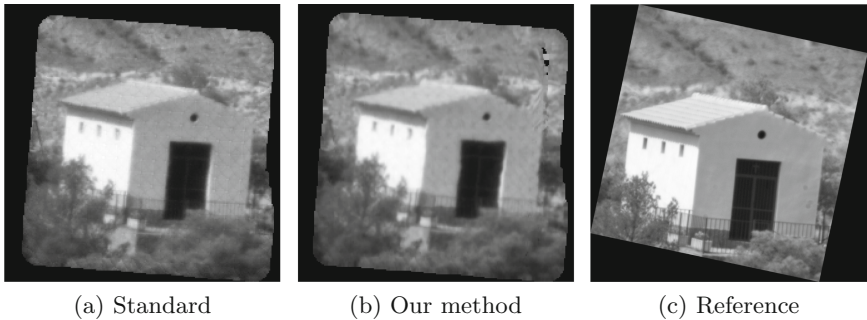
## 4 Results

In order to assess the performance of the plenoptic system, we have captured a set of image sequences in different imaging conditions. Table 1 lists the datasets and the optical parameters of the data. ‘Offset’ is the average offset between the same scene point in successive microlens images and  $m$  is the magnification factor. The numbers refer to different plenoptic camera settings, and the letters indicate different imaging times (i.e. different atmospheric conditions).

**Table 1.** The experimental datasets.

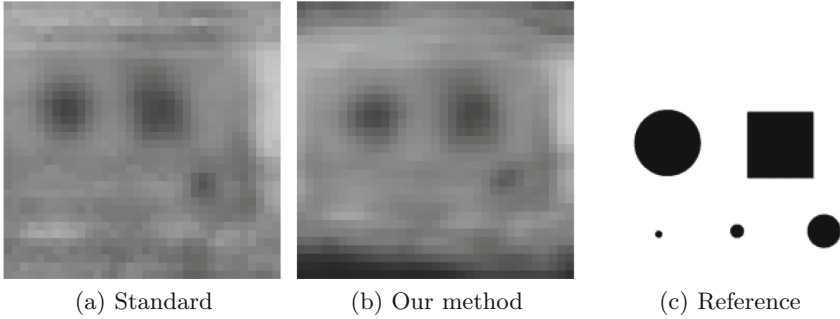
Dataset	Offset (px)	$m$	$a$ (mm)	$b$ (mm)
A0 House	11.75	0.27	5.3	19.8
A1 House	9.5	0.22	5.1	23.2
A2 House	6.0	0.14	4.8	34.2
B0 Target	19.0	0.44	6.1	13.8
B1 Target	8.5	0.20	5.0	25.1
Y1 Target	10.0	0.23	5.2	22.5
A1 Target	9.5	0.22	5.1	23.2

Figure 2 shows the results of reconstruction using a standard reconstruction technique and our method which incorporates distortion, for a single frame of the sequence A1 House, with a reference image for comparison. The image warping is clear from the door edges in (b).



**Fig. 2.** Comparison of methods on A1 House image.

Figure 3 shows the results on the heavily distorted sequence ‘Y1 Target’. This sequence uses artificial heat-generated turbulence. The image is severely distorted in the microlens image and the standard method reconstructs distorted shapes. Our method compensates effectively for the distortion.



**Fig. 3.** Comparison of methods on Y1 Target image.

In order to provide an objective comparison of the reconstruction method, we use sharp edges visible in all the datasets to give an estimate of the image resolution. The blur is computed by fitting a Gaussian convolved with a step function to the edge profile in the images. The Gaussian width  $\sigma$  gives an indication of the reconstruction quality and is listed in Table 2. Application of our method improves the sharpness relative to the standard reconstruction substantially in four of the datasets. The method is more successful at lower magnification parameters.

**Table 2.** Comparison of line spreads between the two methods.

Dataset	Scale factor ( $1/m$ )	Standard	Our method
A0 House	3.7	$5.4 \pm 0.1$	$5.7 \pm 0.1$
A1 House	4.3	$7.6 \pm 0.2$	$5.3 \pm 0.1$
A2 House	7.2	$9.0 \pm 0.1$	$8.0 \pm 0.1$
B0 Target	2.3	$3.3 \pm 0.1$	$3.4 \pm 0.1$
B1 Target	5.1	$3.9 \pm 0.2$	$3.1 \pm 0.2$
Y1 Target	4.8	$8.7 \pm 0.5$	$8.7 \pm 0.2$
A1 Target	4.1	$4.7 \pm 0.1$	$2.9 \pm 0.2$

## 5 Conclusion

In this paper we described a method for inferring reconstructed images from plenoptic camera data, where the images are affected by atmospheric turbulence. The method exploits a Gaussian process to model a smooth image flow field and a linear least squares method to find a consistent reconstruction.

We have collected data with a plenoptic camera and used it to verify our methods. We showed that the algorithms can correctly reconstruct the image and, under more challenging imaging conditions, out-performs a standard reconstruction method.

**Acknowledgment.** This work was supported by DSTL under the CDE programme, grant DSTLX-1000095992R.

## References

1. Kolmogorov, A.N.: Dissipation of energy in locally isotropic turbulence. In: Doklady Akademii Nauk SSSR, vol. 32, p. 16 (1941)
2. Shack, R.V.: Production and use of a lenticular Hartmann screen. *J. Opt. Soc. Am.* **61**(5), 656 (1971)
3. Isaksen, A., McMillan, L., Gortler, S.J.: Dynamically reparameterized light fields. In: SIGGRAPH 2000, pp. 297–306 (2000)
4. Adelson, E.H., Wang, J.Y.A.: Single lens stereo with a plenoptic camera. *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(2), 99–106 (1992)
5. Lumsdaine, A., Georgiev, T.: The focused plenoptic camera. In: Proceedings International Conference on Computational Photography (2009)
6. Mackay, C.D., Baldwin, J., Law, N., Warner, P.: High resolution imaging in the visible from the ground without adaptive optics: new techniques and results. *Proc. SPIE* **5492**, 128 (2004)
7. Labeyrie, A.: Attainment of diffraction limited resolution in large telescopes by fourier analysing speckle patterns in star images. *Astron. Astrophys.* **6**, 85 (1970)
8. Wu, C., Ko, J., Davis, C.C.: Imaging through turbulence using a plenoptic sensor. In: Proceedings of the SPIE 9614, Laser Communication and Propagation through the Atmosphere and Oceans IV, p. 961405 (2015)
9. Wu, C., Ko, J., Davis, C.C.: Object recognition through turbulence with a modified plenoptic camera. In: Proceedings of the SPIE 9354, Free-Space Laser Communication and Atmospheric Propagation XXVII (2015)
10. Koenderink, J.J., Pont, S.C., van Doorn, A.J., Kappers, A.M., Todd, J.T.: The visual light field. *Perception* **36**(11), 1595–1610 (2007)





# Weighted Local Mutual Information for 2D-3D Registration in Vascular Interventions

Cai Meng<sup>1,2(✉)</sup>, Qi Wang<sup>1</sup>, Shaoya Guan<sup>3</sup>, and Yi Xie<sup>1</sup>

<sup>1</sup> School of Astronautics, Beihang University, Beijing 100191, China  
Tsai@buaa.edu.cn

<sup>2</sup> Beijing Advanced Innovation Center for Biomedical Engineering,  
Beihang University, Beijing 100083, China

<sup>3</sup> School of Mechanical Engineering and Automation,  
Beihang University, Beijing 100191, China

**Abstract.** In this paper, a new similarity measure, WLMI (Weighted Local Mutual Information), based on weighted patch and mutual information is proposed to register the preoperative 3D CT model to the intra-operative 2D X-ray images in vascular interventions. We embed this metric into the 2D-3D registration framework, where we show that the robustness and accuracy of the registration can be effectively improved by adapting the strategy of local image patch selection and the weighted joint distribution calculation based on gradient. Experiments on both synthetic and real X-ray image registration show that the proposed method produces considerably better registration results in a shorter time compared with the conventional MI and Normalized MI methods.

**Keywords:** 2D-3D registration · Mutual information · Local patch  
Gradient weighted

## 1 Introduction

The current vascular intervention is usually guided by X-ray image. X-ray image guided intervention, such as digital subtraction angiography (DSA) guided intervention, can track the position of the focus and the surgical instruments in real time, but there is a problem of overlapping between the lesion vessel and the peripheral vessels. While 3D vessel imaging can display lesions from multiple angles, making it easier for doctors to observe and diagnose them. To use 3D data for interventional surgery, we need to register the intra-operative 2D X-ray image and preoperative 3D CT data, that is, 2D-3D registration.

The purpose of 2D-3D vessel registration is to find a transformation parameter that can align the 3D vessel model with the fixed X-ray image after the parameter transformation. Feature-based registration methods generally need

---

Thanks the support by Key projects of NSFC with Grant no. 61533016.

to segment the target object firstly and then register the two point sets [1]. Learning based methods use neural network to evaluate the similarity measure of two images [2] or directly predict the transformation parameters of registration [3]. The intensity based registration method utilizes the pixel intensity information of the entire image and does not require image segmentation. The mutual information (MI) [4] measures the strength of the statistical relationship between two images using their joint probability distribution. What is more, it is widely used in multimodal medical image registration because of its ability to adapt to images of different modalities. However, the global MI measure easily falls into wrong local extremum, and spatial information is completely lost [5].

In order to enhance the robustness of its registration, a lot of improved algorithms based on MI are proposed, such as optimizing the calculation of joint distribution [6–8], combining MI with other common intensity based measures [9]. Because MI only calculates the gray value of each pixel and does not take into account spatial characteristics, the most common improvement is to combine MI with spatial information [10–12]. Although improved methods generally have high registration accuracy, most of them are designed for specific medical images or surgical procedures, which are not applicable to vessel images in vessel interventions. On the one hand, the diffusion of the contrast agent leads to the obvious shadow of the kidney and other parts, whose gray value is similar to vessel. Therefore, the calculation of MI over whole image increases a large number of useless interference information, which has a negative influence on the result. On the other hand, the contrast agent flows with the blood stream, causing parts of vessels to be undeveloped in the image (we call it as vessel excalation), and the extraction of features and edges is inaccurate. So the method of calculating MI at specific feature points is also not applicable. To improve the accuracy of 2D-3D registration during vascular interventional surgery, it is necessary to propose a new similarity measure focusing on the characteristics of vessel images. Furthermore, it is essential to reduce computation complexity by using the information of the vessels.

In this paper we present a new weighted local normalized mutual information measure. According to gradient information and specific selection strategy, the local image patches are extracted and the gradient related weights are used to calculate the NMI value. Desirable results are obtained in the registration experiment of synthetic and real images. The advantages of the proposed WLMI measure can be summarized in the following points:

- Extracting the mask image eliminates most of the unrelated background points in the vessel X-ray image and retain the shape feature of the vessel.
- Obtaining the mask image only uses the information of the fixed image and only needs to be calculated once, which decreases the quantity of calculation.
- In actual registration, the proposed method can avoid the effect of vessel excalation on the registration result, because only the feature in DSA image is extracted and other possible features in the moving image are ignored.

The remainder of this paper is as follows: Sect. 2 describes the proposed similarity measure WLMI, including the method of feature patch extraction, and the

calculation of local mutual information. Section 3 is experimental part, in which we compare the performance of the proposed method with the conventional MI and NMI methods for registration of synthetic X-ray images and real images, followed by our conclusion given in Sect. 4.

## 2 Method

### 2.1 Mutual Information and Normalized Mutual Information

Mutual information (MI) is a basic concept in information theory, used to measure the statistical independence of two random variables or the amount of information that one variable contains another. For vessel registration, the intra-operative X-ray image is defined as the fixed image  $F$ , Digitally Reconstructed Radiograph (DRR) image transformed by the 3D vessel model as the floating image  $M$ . The mutual information of the two is calculated by following:

$$I_{MI}(F, M) = H(F) + H(M) - H(F, M) \quad (1)$$

where  $H(F)$ ,  $H(M)$  is the marginal entropy of  $F$ ,  $M$  respectively,  $H(F, M)$  is the joint entropy that calculated according to the joint probability distribution of two images, defined as:

$$H(F, M) = \sum_{f,m} -P_{F,M}(f, m) \log P_{F,M}(f, m) \quad (2)$$

where the joint probability distribution  $P_{F,M}(f, m)$  can be estimated using joint histograms  $h(f, m)$ . The joint histogram  $h(f, m)$  can be estimated by counting the number of times the intensity pair  $(f, m)$  occurs in the same position of two images, and then the joint distribution probability is estimated by the normalization of the histogram:

$$P_{F,M}(f, m) = \frac{h(f, m)}{\sum_{f,m} h(f, m)} \quad (3)$$

When the two images are correctly matched, MI reaches maximum. Since MI is sensitive to the size of overlapped parts, more robust Normalized Mutual Information [13] (NMI) measure was introduced as

$$I_{NMI}(F, M) = \frac{H(F) + H(M)}{H(F, M)} \quad (4)$$

In DSA images, the complex background may include unrelated information such as the kidney and spine, which will cause a certain interference to vessel registration. Furthermore, vessel exclamation also causes the difficulty of registration. In view of the above two points, the weighted local mutual information is proposed as a new similarity measure.

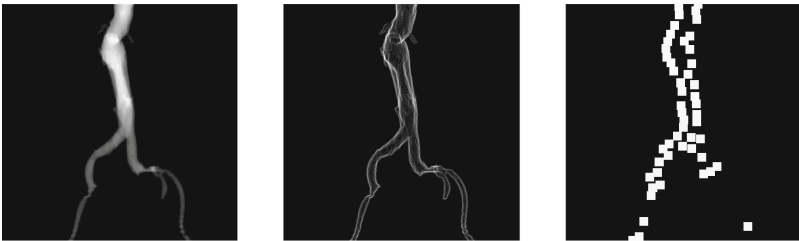
### 2.2 Weighted Local Mutual Information (WLMI) Measure

The weighted local mutual information proposed in this paper is the combination of gradient information and NMI. The gradient information of the fixed image  $F$  is used to filtrate the local patches to get the mask image, and served as the weight of the image patch to estimate the joint distribution histogram.

The generation of mask image  $Mask$  depends on the information of the fixed image only. All points in  $Mask$  are initialized in the state of inactivation. The gradient magnitude  $g(\mathbf{p})$  of each pixel is calculated by Eq. 5, where  $g_x, g_y$  are the gradient along  $X, Y$  axis. Taking each pixel as the center and generating a square window with a side length  $r$ , the area in the window is defined as the “neighborhood patch”  $L_r(\mathbf{p})$ . So each pixel in the fixed image has two characteristics: gradient magnitude  $g(\mathbf{p})$  and neighborhood patch  $L_r(\mathbf{p})$ . Pixels are sorted according to  $g(\mathbf{p})$  from large to small and then retrieved. If the overlap of  $L_r(\mathbf{p})$  and active region in  $Mask$  is less than 20% of the patch size (the overlap equals 0 in the initial state), it is considered that the area is effectively extracted, and  $L_r(\mathbf{p})$  in the Mask is activated. The judgement of overlap is expressed by Eq. 6, where  $Area(\cdot)$  means the number of pixels contained in  $L_r(\mathbf{p})$ . Repeat the above procedure until  $K$  active regions are selected in  $Mask$ . As shown in the following figure, Fig. 1(a) is a vessel DRR image generated by the Ray-casting algorithm [14] based on the CT model. Figure 1(b) is the corresponding gradient map displayed in  $[0, 255]$ , and Fig. 1(c) is the mask image made up of  $K$  neighborhood patches selected according to the gradient value and overlapping principle.

$$g(\mathbf{p}) = \sqrt{g_x^2 + g_y^2} \tag{5}$$

$$L_r(\mathbf{p}) \cap Mask < 20\% \cdot Area(L_r(\mathbf{p})) \tag{6}$$



**Fig. 1.** Images in the process of mask generation. Left is DRR image of vessel. Middle is the corresponding gradient map. Right is mask image (the white part is active area, with parameter  $r = 19, K = 50$ ).

After obtaining the mask image  $Mask$ , NMI can be calculated based on the active region. When the joint distribution histogram is counted, the pixels within the active region are considered only in  $F$  and  $M$ , then the joint distribution

probability is estimated and the NMI value is calculated. We defined this similarity calculation as Local mutual information (LMI). In LMI, the joint distribution histogram is obtained by counting the number of times the intensity pair  $(f, m)$  occurs in the same position of two images, which means that the intensity pair in each position contributes equally to the histogram. The weight is expressed by

$$w_{LMI}(\mathbf{p}) = \begin{cases} 1, & Mask(\mathbf{p}) \neq 0 \\ 0, & else \end{cases} \quad (7)$$

To distinguish the importance of different gradient positions to the registration results, we propose to give a weight  $w(\mathbf{p})$  to each patch  $L_r(\mathbf{p})$  to represent the effect on the registration. The weight  $w(\mathbf{p})$  is positively correlated with the gradient  $g(\mathbf{p})$ , calculated by

$$w_{WLMI}(\mathbf{p}) = \begin{cases} \frac{g(\mathbf{p})}{g(\mathbf{p}) + 1}, & L_r(\mathbf{p}) \text{ is active} \\ 0, & else \end{cases} \quad (8)$$

Each pixel in patch  $L_r(\mathbf{p})$  shares the same weight  $w(\mathbf{p})$  in *mask*. When calculating the joint distribution histogram, the number of pixels are replaced by the sum of weights of these pixels. The addition of weights adjusts the shape of the joint distribution histogram  $h(f, m)$  and changes the joint distribution probability  $P(f, m)$ . Then the Eqs. 3, 2 and 4 are used to calculate WLMI as the final measure value. The calculation procedure of WLMI is shown in Fig. 2. The process of obtaining the mask image is equivalent to extracting feature of the fixed image, and then using the feature to estimate the registration degree.

WLMI is incorporated in the 2D-3D registration framework. First, 3D vessel model is converted into a 2D DRR image under specific transformation parameter  $T$ ; then WLMI value of DRR and X-ray images are calculated to determine the quality of registration; finally, Powell algorithm is utilized to generate new transformation parameter  $T_{new}$  and iteratively optimizes the transformation parameter until the WLMI is maximized.

### 3 Experiments and Results

#### 3.1 Experiment Setup

In the registration experiment we evaluate our method on a patient's computed tomography angiography (CTA) consisting of 126 DICOM images. The size of 3D image is  $512 \times 512 \times 126$  with a pixel spacing of  $0.68 \times 0.68 \times 5.0$  mm. Reconstruct the CTA image and threshold segmentation is adopted to segment vessel. Use the vessel model to generate DRR images mimicking the rigid geometry of the X-ray imaging, with dimension  $512 \times 512$  and pixel spacing  $1 \times 1$  mm. In order to resemble the real intra-operative X-ray image, the DRR images are processed according to Eq. 9 to generate synthetic X-ray images:

$$I = \mu \cdot I_{bg} + \gamma \cdot G_{\sigma} * I_{DRR} + N(a, b) \quad (9)$$

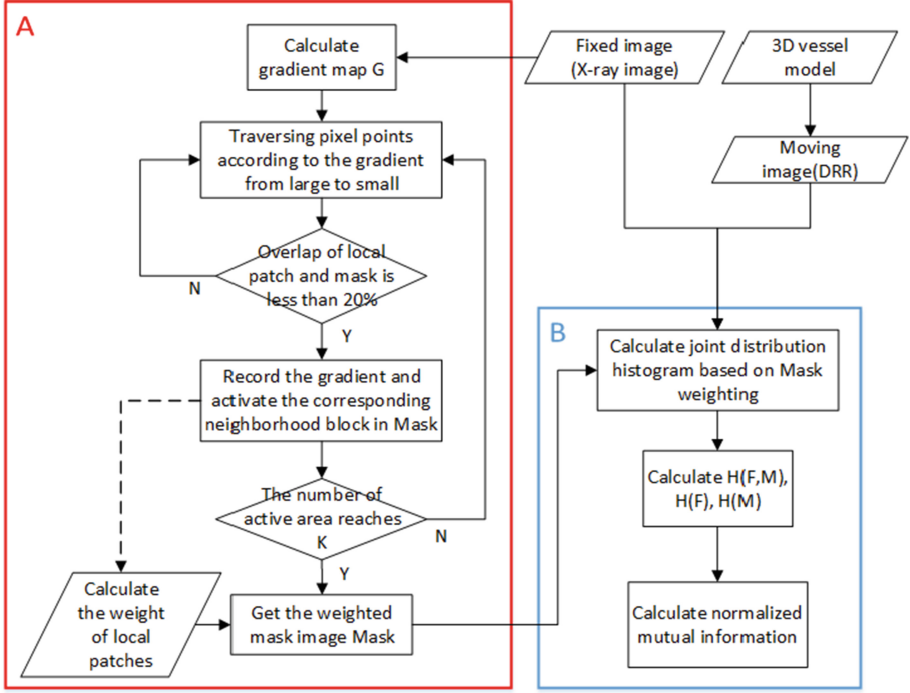


Fig. 2. The calculation procedure of WLMI

where  $I_{bg}$  is the background that picked from the real X-ray images of the vascular interventions,  $I_{DRR}$  is the DRR image,  $G_\sigma$  is a Gaussian smoothing kernel with a standard deviation  $\sigma$  simulating X-ray scattering effect,  $N(a, b)$  is a random noise uniformly distributed on  $[a, b]$ , and  $(\mu, \gamma)$  are synthetic coefficients. We found that setting  $(\mu, \gamma, \sigma, a, b) = (0.6, 0.8, 0.5, -5, 5)$  can get the synthetic images closest to real images.

Without considering elastic deformation, The transformation parameter in 3D space are six degrees of freedom, which can be expressed as  $T = \{r_x, r_y, r_z, t_x, t_y, t_z\}$ .  $(t_x, t_y, t_z)$ ,  $(r_x, r_y, r_z)$  are the relative translation and rotation along/around each of the standard axes, in which the translation along Z axis  $t_z$  is equivalent to image scaling. The accuracy of registration is generally measured by the mean Target Registration Error in the direction of the projection ( $mTREproj$ ) and the mean absolute error ( $MAE$ ) of each registration parameter. The  $mTREproj$  and  $MAE$  are defined as following:

$$mTREproj = \frac{1}{N} \sum_{n=1}^N (T \circ P_n - \hat{T} \circ P_n) \tag{10}$$

$$MAE_i = |T_i - \hat{T}_i|, i \in [1, 6] \tag{11}$$

where  $N$  is the number of points selected in 2D CTA image,  $P_n$  is the  $n$ -th point,  $T$  and  $\hat{T}$  are the true and estimated transformation respectively.

### 3.2 Intact Vessel Registration

The proposed method is implemented by MATLAB, the DRR generation part is implemented by ITK. 10 experiments are carried out to verify the validity of WLMI. The initial registration parameters are randomly generated in the range of  $\pm 10$  mm and  $\pm 10^\circ$ . The comparison method is LMI, traditional MI and NMI measurement. Figure 4(a) summaries the statistics of the  $MAE$  in each transformation parameter. Table 1 shows the  $mTREproj$  and registration time. The results show that WLMI and LMI have higher registration accuracy and shorter registration time than the traditional MI and NMI. WLMI has better convergence effect than LMI in the parameter  $t_z$  which representing the zoom effect, and the registration results are more stable.

**Table 1.** Comparison of  $mTREproj$  and registration time under vessel intactness

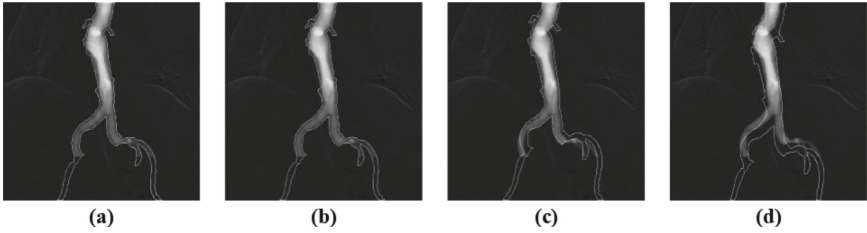
	WLMI	LMI	NMI	MI
$mTREproj$ (mm)	2.4	6.4	22.7	24.7
Time/iteration (s)	190.5	202.3	243.9	240.6

### 3.3 Excalate Vessel Registration

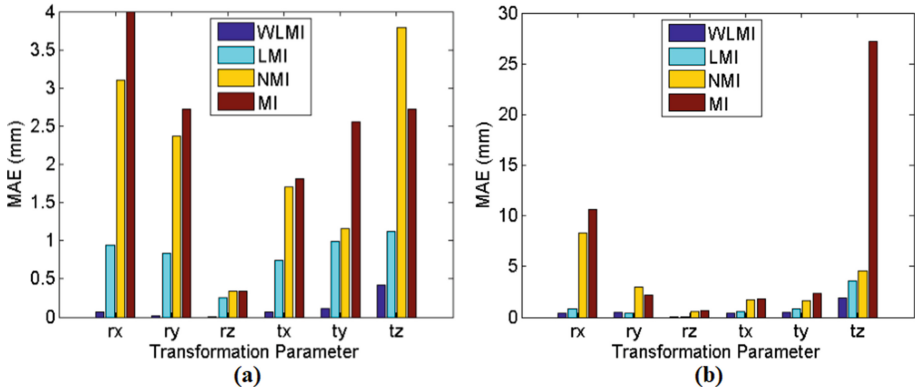
For the second experiment, We want to verify the robustness of the method by registration of the vessel excalation. The experiments are conducted under the same condition with intact vessel registration experiment. Figure 3 is a superimposed display of registration results and fixed images. Figure 4(b) and Table 2 are the statistical results of registration error  $MAE$ ,  $mTREproj$  and registration time. The results show that WLMI and LMI measures based on feature patches are less susceptible to vascular loss than NMI and MI measures, allowing faster and more accurate registration results.

### 3.4 Real Vessel Images Registration

In the third experiment, real vessel registration experiment was conducted on patients' CTA and DSA images in the real operating environment. The size of 3D image is  $512 \times 512 \times 139$  with a pixel spacing of  $0.68 \times 0.68 \times 5.0$  mm. The size of DSA image is  $1024 \times 1024$  with a pixel spacing of  $0.37 \times 0.37$  mm. We selected one of the 244 DSA sequences generated from once injection of contrast agent as the fixed image of registration. The initial transformation parameter is estimated according to the position of C arm and CT machine. Figure 5 shows the 2D-3D registration results of real vessel image with WLMI as measurement. Though the real image registration does not have a gold standard registration parameter, it can be seen from the figure that the WLMI registration result has the basical same vessel contour as the real DSA image.



**Fig. 3.** The registration result of WLMI, LMI, NMI, MI under vessel excalation. The white contour line is the vessel boundary in the DRR image corresponding to the registration result parameter.



**Fig. 4.** (a) Comparison of MAE under vessel intactness, (b) comparison of MAE under vessel excalation

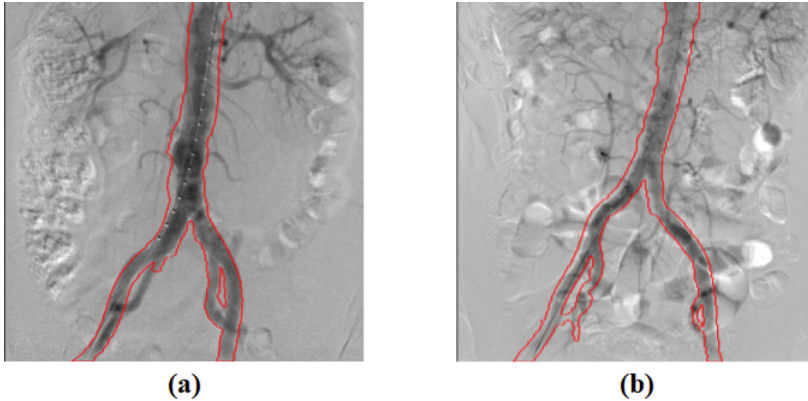
**Table 2.** Comparison of *mTREproj* and registration time under vessel excalation

	WLMI	LMI	NMI	MI
<i>mTREproj</i> (mm)	2.9	6.4	37.1	42.2
Time/iteration (s)	188.4	179.4	237.2	241.6

### 3.5 Discussion

The WLMI method proposed in this paper is more effective and faster than the traditional method in the registration of vascular interventions. Compared with the traditional NMI, the WLMI measure curve has bigger gradient in the same situation, so it is easier to converge. However, due to the extraction of local image patches, the performance of WLMI measurement on smoothness is not as good as expected, which is easy to fall into local extremum. Therefore, the selection of optimization methods and the adjustment of parameters are more sensitive than NMI. How to improve the smoothness and stability of WLMI measurement is the focus of the next study.





**Fig. 5.** The real vessel image registration result of WLMI. The red contour line is the edge of vessel in DRR corresponding to the registration result parameter, and the background is the real vessel DSA image. (Color figure online)

In addition, for the registration of real vessel images, the accuracy of vessel segmentation when generating 3D models, the sharpness and contrast of vessels in the DSA images, will all affect the final registration results. These influencing factors are also issues that need further study.

## 4 Conclusion

This paper presents a new similarity measure WLMI for the registration of pre-operative CT images and intraoperative X-ray images in vascular interventions. The positions of local area are determined based on the gradient information of fixed image, and the local image patches are extracted from the fixed image and the floating image respectively to calculate the weighted normalized mutual information, thereby evaluating the similarity of the two images and performing 2D-3D registration. The experiments of vessel intactness and Excalation were conducted on synthetic X-ray images. The results show that the proposed WLMI measure has faster and more accurate registration effect.

## References

1. Duong, L., Liao, R., Sundar, H., Tailhades, B., Meyer, A., Xu, C.: Curve-based 2D-3D registration of coronary vessels for image guided procedure. In: International Society for Optics and Photonics, Medical Imaging 2009: Visualization, Image-Guided Procedures, and Modeling, vol. 7261, pp. 72610S (2009)
2. Simonovsky, M., Gutiérrez-Becker, B., Mateus, D., Navab, N., Komodakis, N.: A deep metric for multimodal registration. In: Ourselin, S., Joskowicz, L., Sabuncu, M.R., Unal, G., Wells, W. (eds.) MICCAI 2016. LNCS, vol. 9902, pp. 10–18. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-46726-9\\_2](https://doi.org/10.1007/978-3-319-46726-9_2)

3. Miao, S., Wang, Z.J., Zheng, Y., Liao, R.: Real-time 2D/3D registration via CNN regression. In: 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI), pp. 1430–1434. IEEE (2016)
4. Roche, A., Malandain, G., Pennec, X., Ayache, N.: The correlation ratio as a new similarity measure for multimodal image registration. In: Wells, W.M., Colchester, A., Delp, S. (eds.) MICCAI 1998. LNCS, vol. 1496, pp. 1115–1124. Springer, Heidelberg (1998). <https://doi.org/10.1007/BFb0056301>
5. Shadaydeh, M., Sziranyi, T.: An improved mutual information similarity measure for registration of multi-modal remote sensing images. In: International Society for Optics and Photonics, Image and Signal Processing for Remote Sensing XXI, vol. 9643, pp. 96430F (2015)
6. Xuesong, L., Zhang, S., He, S., Chen, Y.: Mutual information-based multimodal image registration using a novel joint histogram estimation. *Comput. Med. Imaging Graph.* **32**(3), 202–209 (2008)
7. Rubeaux, M., Nunes, J.-C., Albera, L., Garreau, M.: Edgeworth-based approximation of mutual information for medical image registration. In: 2010 2nd International Conference on Image Processing Theory Tools and Applications (IPTA), pp. 195–200. IEEE (2010)
8. Pradhan, S., Patra, D.: Enhanced mutual information based medical image registration. *IET Image Proc.* **10**(5), 418–427 (2016)
9. Andronache, A., von Siebenthal, M., Székely, G., Cattin, P.: Non-rigid registration of multi-modal images using both mutual information and cross-correlation. *Med. Image Anal.* **12**(1), 3–15 (2008)
10. Legg, P.A., Rosin, P.L., Marshall, D., Morgan, J.E.: Feature neighbourhood mutual information for multi-modal image registration: an application to eye fundus imaging. *Pattern Recogn.* **48**(6), 1937–1946 (2015)
11. Russakoff, D.B., Tomasi, C., Rohlfing, T., Maurer, C.R.: Image similarity using mutual information of regions. In: Pajdla, T., Matas, J. (eds.) ECCV 2004. LNCS, vol. 3023, pp. 596–607. Springer, Heidelberg (2004). [https://doi.org/10.1007/978-3-540-24672-5\\_47](https://doi.org/10.1007/978-3-540-24672-5_47)
12. Luan, H., Qi, F., Xue, Z., Chen, L., Shen, D.: Multimodality image registration by maximization of quantitative-qualitative measure of mutual information. *Pattern Recogn.* **41**(1), 285–298 (2008)
13. Studholme, C., Hill, D.L.G., Hawkes, D.J.: An overlap invariant entropy measure of 3D medical image alignment. *Pattern Recogn.* **32**(1), 71–86 (1999)
14. Kruger, J., Westermann, R.: Acceleration techniques for GPU-based volume rendering. In: Proceedings of the 14th IEEE Visualization 2003 (VIS 2003), p. 38. IEEE Computer Society (2003)



# Cross-Model Retrieval with Reconstruct Hashing

Yun Liu<sup>1</sup>, Cheng Yan<sup>2</sup>(✉), Xiao Bai<sup>2</sup>, and Jun Zhou<sup>3</sup>

<sup>1</sup> School of Automation Science and Electrical Engineering,  
Beihang University, Beijing, China  
liuyun@buaa.edu.cn

<sup>2</sup> School of Computer Science and Engineering,  
Beihang University, Beijing, China  
{beihangyc, baixiao}@buaa.edu.cn

<sup>3</sup> School of Information and Communication Technology,  
Griffith University, Nathan, Australia  
jun.zhou@griffith.edu.au

**Abstract.** Hashing has been widely used in large-scale vision problems thanks to its efficiency in both storage and speed. For fast cross-modal retrieval task, cross-modal hashing (CMH) has received increasing attention recently with its ability to improve quality of hash coding by exploiting the semantic correlation across different modalities. Most traditional CMH methods focus on designing a good hash function to use supervised information appropriately, but the performance are limited by hand-crafted features. Some deep learning based CMH methods focus on learning good features by using deep network, however, directly quantizing the feature may result in large loss for hashing. In this paper, we propose a novel end-to-end deep cross-modal hashing framework, integrating feature and hash-code learning into the same network. We keep the relationship of features between modalities. For hash process, we design a novel net structure and loss for hash learning as well as reconstruct the hash codes to features to improve the quality of codes. Experiments on standard databases for cross-modal retrieval show the proposed methods yields substantial boosts over latest state-of-the-art hashing methods.

## 1 Introduction

Nearest neighbor (NN) search has been widely adopted in image retrieval. The time complexity of the NN method on a dataset of size  $n$  is  $O(n)$ , which is infeasible for real-time retrieval on large dataset, especially multimedia big data with large volumes and high dimensions. Approximate nearest neighbor (ANN) search has been proposed to make NN search scalable, and becomes a preferred solution in many computer vision and machine learning applications [6, 8, 18, 25, 27]. The goal of ANN search is to find approximate results rather than exact ones so as to achieve high speed data processing [10, 22]. Amongst various ANN search techniques, hashing is widely studied because of its efficiency in both

storage and speed. By generating binary codes for image data, the retrieval on a dataset with millions of samples can be completed in a constant time using only tens of hash bits [9, 16, 28, 30, 33, 34].

In many applications, the data have not only one modality such as image-text. Many social websites and Flickr have image data with corresponding text information such as tags. These data having at least two types information are called multi-modal data. With the rapid growth of multi-modal data, it is important to encode these data for cross-modal retrieval which returns semantic relevant results of one modality with respect to a query in the other modality. Hashing, as a promising solution, can be used to handle the cross-modal retrieval task. Cross-modal hashing can transform high-dimensional data into binary codes and keep the similarity of each sample in binary codes for fast search.

Many cross-modal hashing methods [3, 7, 12, 14, 23, 26, 31, 32, 35, 36] have been proposed to capture correlation structures of data in different modalities and index the cross-modal data into binary codes to ensure the similar data in Hamming space having a small distance. Generally, they can be divided into two types: unsupervised methods [14, 26, 35] and supervised methods [2, 12, 29, 36]. These unsupervised methods generally focus on keeping the distribution of original data in new Hamming space that can be trained without labels. However, they are limited by the semantic gap dilemma. The low-level feature descriptors can not reflect the high-level semantic information of an object, and the relationship of each other is hard to capture. Supervised cross-modal hashing methods generally focus on indexing the cross-modal data to binary codes with corresponding labels or relevance feedbacks to relieve the semantic gap for better hashing quality such as high performance with short codes.

Some of these supervised cross-modal hash methods use hand-crafted features to exploit shared structures across different modalities for hashing process. The feature extraction procedure is independent of the hashing process. Though the hashing process is well designed, the feature might not be compatible, which is a shortcoming of these methods. Hence, they can not achieve approving performance. With the development of deep learning technique, the neural networks has been widely used for feature learning. More and more deep framework hash methods [2, 15, 17, 19, 21, 37] are proposed to achieve binary codes with higher quality for retrieval task. Cross-model deep hash methods [12] focus on learning features preserving the correlation of samples in different modalities and combining a hash codes learning process to minimize the quantization loss, however, directly quantizing the feature may affect the quality of hash codes.

In this work, we propose a novel deep learning methods for cross-modal hashing. It is an end-to-end learning framework. Different from previous work that just use correlation information for feature learning part, we not only consider semantic relationship in the loss function for hash learning but also reconstruct the hash codes for better performance. The main contributions are outlined as follows:

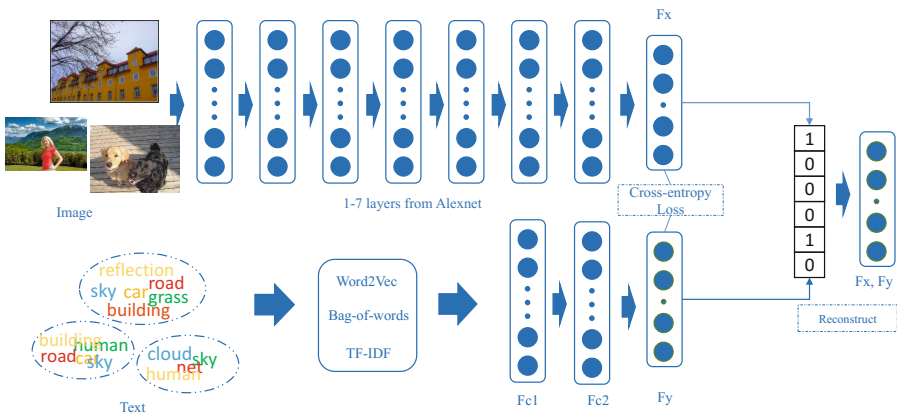
- It is a novel end-to-end learning framework integrating feature learning and hash learning into the same net to guarantee the code quality.

- Correlation and reconstruct loss are designed for whole net training to guaranteed the quality of hash codes.
- Experiments on real image-text modalities databases show that our method achieve the state-of-the-art performance in cross-modal hashing retrieval applications.

## 2 Method

### 2.1 Model Structure

Our model is an end-to-end deep learning framework for cross-modal retrieval task. For convenience, we separate the network into two parts to explain in detail. As shown in Fig. 1, the first part is from *Image* and *Text* to  $F_x$  and  $F_y$ . This part is to learn the correlation in two modalities, whose target is to ensure  $F_x$  and  $F_y$  for each sample preserving the correlation between modalities to give the second part well inputs. The second part is reconstruct part, which is the rest in Fig. 1. In this part, we reconstruct the hash codes to features  $F_x$  and  $F_y$  to guarantee the quality of codes. Across the whole net, each input data will be given a hash codes finally. We designed a well-specified loss function for capturing the correlations of two modalities. Under guarantees of the learning process, the relationship of each sample can be well preserved by their hash codes. All the learning process and back-propagation are implemented as a whole.



**Fig. 1.** Our method is an end-to-end deep framework with correlation and reconstruct hash learning.

### 2.2 Correlation Feature Learning

In the correlation feature learning part of the framework, there are two pipelines for the image and text modalities. With respect to the image network, we follow

the AlexNet [13], except the last fully connected layer, which is designed as feature layer with short length in our model. The image data can be used as the input after resizing ( $227 * 227 * 3$ ). In the text pipeline, each input is a vector with bag-of-words (BOW) representation. The network is composed of three fully connected layers corresponding to the last three layers of the image network with the same number of nodes. The details about the two pipelines are shown in Table 1. Notice that, the Local Response Normalization (LRN) is used after *conv1* and *conv2*, and the Rectified Linear Unit (ReLU) is used for all of the first seven layers of image net and all of the first two layers of the text net as an activation function.

**Table 1.** Configuration of two pipelines of network, in which  $k$  = kernel,  $s$  = stride,  $p$  = pad,  $pk$  = pooling kernel,  $ps$  = pooling stride

Layer	Configuration
conv1	$k : 96 \times 11 \times 11, s : 4, p : 0, pk : 3, ps : 2$
conv2	$k : 256 \times 5 \times 5, s : 4, p : 2, pk : 3, ps : 2$
conv3	$k : 384 \times 3 \times 3, s : 0, p : 1$
conv4	$k : 384 \times 3 \times 3, s : 0, p : 1$
conv5	$k : 256 \times 3 \times 3, s : 0, p : 1, pk : 3, ps : 2$
fc(img)	img-fc1:4096, img-fc2:4096, $F_x \cdot d$
fc(txt)	Fc1:4096, Fc2:4096, $F_y \cdot d$

Let  $X = \{x_1, x_2, \dots, x_m\}$  denote the inputs of the images, and  $Y = \{y_1, y_2, \dots, y_n\}$  denote the inputs of the texts. Let  $f_x$  and  $f_y$  be the features ( $F_x$  and  $F_y$ ) of image and text of each sample. We use  $S$  as correlation similarity matrix for feature learning, where  $s_{ij} = 0$  if the image  $x_i$  and text  $y_j$  are dissimilar and  $s_{ij} = 1$  otherwise. Note that, the similarity associated with the semantic information, such as label information, which means that if the image and text are similar, they have the same label and if they belong to different categories, they are dissimilar. The purpose of this part is to guarantee the  $f_{x_i}$  and  $f_{y_j}$  capturing the relationship according to similarity labels  $s_{ij}$ . Inspired by [5, 12], we use logarithm Maximum a Posteriori (MAP) estimation for the features  $F_x = [f_{x_1}, f_{x_2}, \dots, f_{x_m}]$  and  $F_y = \{f_{y_1}, f_{y_2}, \dots, f_{y_n}\}$ . The objective function is defined as

$$\log p(F_x, F_y | S^f) \propto \log p(S^f | F_x, F_y) p(F_x) p(F_y) \quad (1)$$

where  $p(F_x)$  and  $p(F_y)$  are prior distributions, and  $p(F_x, F_y | S^f)$  is the likelihood function. It is equal to

$$\max_{i,j} \sum \log p(s_{ij}^f | f_{x_i}, f_{y_j}) p(f_{x_i}) p(f_{y_j}) \quad (2)$$

where  $p(s_{ij}|f_{x_i}, f_{y_j})$  is probability of the relationship between  $x_i$  and  $y_j$ . If  $x_i$  and  $y_j$  are given, we can get it by

$$p(s_{ij}^f|f_{x_i}, f_{y_j}) = \phi(f_{x_i}, f_{y_j})^{s_{ij}}(1 - \phi(f_{x_i}, f_{y_j}))^{1-s_{ij}} \quad (3)$$

where  $\phi(x, y) = -1/(1 + e^{-\alpha x^T \cdot y})$  is the sigmoid function with  $\alpha$  to control the bandwidth, and the  $x^T \cdot y$  is the inner product of vector  $x$  and  $y$ . We can regard it as an extension of the logistic regression classifier. If the label  $s_{ij} = 1$ , the larger of  $f_{x_i}^T \cdot f_{y_j}$ , the larger  $p(s_{ij} = 1|f_{x_i}, f_{y_j})$ , which means the two sample should be similar, and if  $p(s_{ij} = 0|f_{x_i}, f_{y_j})$  is large, the two sample should be dissimilar. When the Eq. 3 is maximized, the feature level relationship  $S$  between different modalities can be preserved in the features  $f_{x_i}$  and  $f_{y_j}$ . Combine with Eqs. 1, 2 and 3, finally, we can get the feature level cross-model loss

$$L_f = \sum_{s_{i,j}} \log(1 + \exp(\alpha f_{x_i}^T \cdot f_{y_j})) - s_{ij} \alpha f_{x_i}^T \cdot f_{y_j} \quad (4)$$

With minimized Eq. 4, if the relationship of two sample is  $s_{ij} = 1$ , the inner product of their features should be large, and if  $s_{ij} = 0$  otherwise.  $\alpha$  is the hyper-parameter to guarantee effective back-propagation for training. Note that, the learning of this part is not just based on Eq. 4. In other words, the gradient of this part in back-propagation process contains the loss of two parts. As part of the whole learning process, it is an assurance for giving the hash learning part good inputs. Though the features keep correlation with each other in some degree, they are not quite fit for binaryzation. So we design a reconstruct hash coding part. Combined with hash learning part, the feature learning part will provide more suitable features for hashing after training.

The reconstruct hashing part is designed to guarantee the quality of codes. When we get the feature of each point, we should binary them. To guarantee the features and hash codes are as similar as possible, we don't just use sign function. The loss is designed as follow

$$L_h = \sum_i \|f_i - Wb_i - c\|^2 + \beta \|f_i - b_i\|^2 + \gamma \|W\|^2 \quad (5)$$

where  $f_i \in \{f_x, f_y\}$  represent one of the features of the data point from both modalities, and  $b_i$  is the corresponding binary codes. When we get the feature  $f_i$  of each point, we use sign function to binary it. The first term of Eq. 5 is the reconstruct term that guarantee the binary codes of each point is similar to its feature when after reconstruct, which is a project of  $b_i$ . The second term is to force the feature and binary codes are as similar as possible, and the third term is a regular term of the project matrix.  $\beta$  and  $\gamma$  are the hyper-parameter to control balance of each term.

**Table 2.** MAPs of different methods for Image-to-Text retrieval.

Dataset	NUS-WIDE			MIR-FLICKR		
	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
IMH [26]	0.433	0.425	0.428	0.552	0.561	0.557
CM-NN [24]	0.601	0.605	0.613	0.723	0.731	0.740
QCH [32]	0.487	0.500	0.512	0.651	0.665	0.671
CorrAE [7]	0.451	0.461	0.494	0.625	0.632	0.643
SCM [36]	0.461	0.467	0.475	0.643	0.645	0.645
SePH [20]	0.475	0.491	0.496	0.635	0.657	0.671
DCMH [12]	0.601	0.667	0.735	0.761	0.786	0.807
Ours	<b>0.773</b>	<b>0.791</b>	<b>0.809</b>	<b>0.800</b>	<b>0.808</b>	<b>0.821</b>

We combine two parts of loss Eqs. 4 and 5 together to get the final loss

$$\begin{aligned}
\min L &= L_f + \lambda L_h \\
&= \sum_{s_{i,j}} \log(1 + \exp(\alpha f_{x_i}^T \cdot f_{y_j})) - s_{ij} \alpha f_{x_i}^T \cdot f_{y_j} \\
&\quad + \lambda \left( \sum_i \|f_i - Wb_i - c\|^2 + \beta \|f_i - b_i\|^2 + \gamma \|W\|^2 \right)
\end{aligned} \tag{6}$$

where  $\lambda$  keeps the balance of  $L_f$  and  $L_h$ . We adopt an alternating learning strategy to learn the parameters. We can efficiently optimize the net parameters via automatic differentiation techniques in Google TensorFlow [1]. For  $b_i$ , when net parameters are fixed, we can sign  $f_i$  to get it.

### 3 Experiment

Our method is implemented with Google TensorFlow [1], and network is trained on a NVIDIA TITAN X 12GB GPU. All of our experiments are finished on image-text databases.

#### 3.1 Database

We use **NUS-WIDE** and **MIR-FLICKR** [11] for experiment.

**MIR-FLICKR** is a dataset with 25k images collected from Flickr website. Each sample is also an image-text pair and we select the samples having at least 20 textual tags for our experiment. All the images are resized to 256 \* 256 \* 3 and the corresponding text is represented as BOW vector with 1386 dimensionality. Each sample is labeled with some of the 24 concepts. For all databases, if point  $x_i$  and  $y_j$  share at least one common label, we consider they are similar. Otherwise, they are considered to be dissimilar.



**Table 3.** MAPs of different methods for Text-to-Image retrieval.

Dataset	NUS-WIDE			MIR-FLICKR		
	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
IMH [26]	0.451	0.443	0.417	0.561	0.560	0.559
CM-NN [24]	0.602	0.622	0.643	0.718	0.721	0.729
QCH [32]	0.515	0.548	0.562	0.638	0.641	0.650
CorrAE [7]	0.451	0.465	0.478	0.612	0.625	0.641
SCM [36]	0.483	0.511	0.524	0.586	0.588	0.601
SePH [20]	0.482	0.490	0.505	0.573	0.590	0.596
DVSH [4]	0.731	0.761	0.773	0.761	0.776	0.779
Ours	<b>0.775</b>	<b>0.785</b>	<b>0.801</b>	<b>0.807</b>	<b>0.815</b>	<b>0.823</b>

**NUS-WIDE** is a multi-label dataset containing more than 260k images, with a total number of 5,018 unique tags. Each image annotated with one or multiple labels from 81 concepts as ground-truth for evaluation. Following prior works [12, 31], we use the subset of the NUS-wide including 195,834 image-text pairs which belong to 21 most frequent concepts of the total concepts. All the images are resized to  $256 * 256 * 3$  and all the text for each sample is represented as a bag-of-words (BOW) vector with 1000 dimensionality.

### 3.2 Compared Methods

For comparison, we adopted eight state-of-the-art cross-modal hashing methods as baselines, including IMH [26], CorrAE [7], SCM [36], CM-NN [24], QCH [32], SePH [20], DCMH [12]. The DCMH is deep cross-modal hash methods proposed recently. The codes of IMH, CorrAE, CM-NN, SePH, DCMH are provided by the corresponding authors. With respect to the rest methods whose codes are not available, we implement them by ourselves.

To evaluate the retrieval performance, we follow [12, 20, 32] to use mean Average Precision (mAP) which is widely used. We adopt  $\text{mAP}@R = 500$ , which is same to [20, 32].

The mAP results for ours and other baselines on *NUS-WIDE* and *MIR-FLICKR* databases are reported in Tables 2 and 3. The experiments results are shown that the our method has better performance than all of the compared methods.

## 4 Conclusion

In this paper, we have proposed a hash based cross-modal method for cross-modal retrieval applications. It is an end-to-end deep learning framework that extract features as well as reconstruct hash codes to guarantee the quality of hash

codes. Experiments on three databases show that our method can outperform other baselines to achieve the state-of-the-art performance in real applications.

**Acknowledgement.** This work was supported by the National Natural Science Foundation of China project No. 61772057, in part by Beijing Natural Science Foundation project No. 4162037, and the support funding from State Key Lab of Software Development Environment.

## References

1. Abadi, M., et al.: TensorFlow: large-scale machine learning on heterogeneous systems (2015). Software: [tensorflow.org](https://www.tensorflow.org)
2. Andrew, G., Arora, R., Bilmes, J., Livescu, K.: Deep canonical correlation analysis. In: ICML, pp. III–1247 (2013)
3. Bronstein, M.M., Bronstein, A.M., Michel, F., Paragios, N.: Data fusion through cross-modality metric learning using similarity-sensitive hashing. In: CVPR, pp. 3594–3601 (2010)
4. Cao, Y., Long, M., Wang, J., Yang, Q., Yu, P.S.: Deep visual-semantic hashing for cross-modal retrieval. In: SIGKDD, pp. 1445–1454 (2016)
5. Cao, Z., Long, M., Yang, Q.: Transitive hashing network for heterogeneous multimedia retrieval. In: AAAI
6. Carreira-Perpinan, M.A., Razi-perchikolaei, R.: Hashing with binary autoencoders. In: CVPR, pp. 557–566 (2015)
7. Feng, F., Wang, X., Li, R.: Cross-modal retrieval with correspondence autoencoder. In: MM, pp. 7–16 (2014)
8. Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F.: Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. TPAMI **35**(12), 2916–2929 (2013)
9. Yang, H., et al.: Maximum margin hashing with supervised information. MTAP **75**, 3955–3971 (2016)
10. Heo, J.P., Lee, Y., He, J., Chang, S.F.: Spherical hashing. In: CVPR, pp. 2957–2964 (2012)
11. Huiskes, M.J., Lew, M.S.: The MIR flickr retrieval evaluation. In: SIGIR, pp. 39–43 (2008)
12. Jiang, Q.Y., Li, W.J.: Deep cross-modal hashing. In: CVPR, pp. 3232–3240 (2017)
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS, pp. 1097–1105 (2012)
14. Kumar, S., Udupa, R.: Learning hash functions for cross-view similarity search. In: IJCAI, pp. 1360–1365 (2011)
15. Lai, H., Pan, Y., Liu, Y., Yan, S.: Simultaneous feature learning and hash coding with deep neural networks. In: CVPR, pp. 3270–3278 (2015)
16. Zhou, L., Bai, X., Liu, X., Zhou, J.: Binary coding by matrix classifier for efficient subspace retrieval. In: ICMR, pp. 82–90 (2018)
17. Li, W.J., Wang, S., Kang, W.C.: Feature learning based deep supervised hashing with pairwise labels. In: IJCAI, pp. 1711–1717 (2016)
18. Lin, G., Shen, C., Shi, Q., Van den Hengel, A., Suter, D.: Fast supervised hashing with decision trees for high-dimensional data. In: CVPR, pp. 1971–1978 (2014)
19. Lin, J., Li, Z., Tang, J.: Discriminative deep hashing for scalable face image retrieval. In: IJCAI, pp. 2266–2272 (2017)

20. Lin, Z., Ding, G., Hu, M., Wang, J.: Semantics-preserving hashing for cross-view retrieval. In: CVPR, pp. 3864–3872 (2015)
21. Liong, V.E., Lu, J., Wang, G., Moulin, P., Zhou, J.: Deep hashing for compact binary codes learning. In: CVPR, pp. 2475–2483 (2015)
22. Liu, W., Wang, J., Ji, R., Jiang, Y.-G., Chang, S.-F.: Supervised hashing with kernels. In: CVPR, pp. 2074–2081 (2012)
23. Liu, X., He, J., Deng, C., Lang, B.: Collaborative hashing. In: CVPR, pp. 2147–2154 (2014)
24. Masci, J., Bronstein, M.M., Bronstein, A.M., Schmidhuber, J.: Multimodal similarity-preserving hashing. TPAMI **36**(4), 824–830 (2014)
25. Shen, F., Shen, C., Shi, Q., Van den Hengel, A., Tang, Z.: Inductive hashing on manifolds. In: CVPR, pp. 1562–1569 (2013)
26. Song, J., Yang, Y., Yang, Y., Huang, Z., Shen, H.T.: Inter-media hashing for large-scale retrieval from heterogeneous data sources. In: SIGMOD, pp. 785–796 (2013)
27. Strecha, C., Bronstein, A.M., Bronstein, M.M., Fua, P.: LDAHash: improved matching with smaller descriptors. TPAMI **34**(1), 66–78 (2012)
28. Torralba, A., Fergus, R., Weiss, Y.: Small codes and large image databases for recognition. In: CVPR, pp. 1–8 (2008)
29. Wang, D., Gao, X., Wang, X., He, L.: Semantic topic multimodal hashing for cross-media retrieval. In: AAAI, pp. 3890–3896 (2015)
30. Wang, J., Kumar, S., Chang, S.-F.: Semi-supervised hashing for large-scale search. TPAMI **34**(12), 2393–2406 (2012)
31. Wang, W., Ooi, B.C., Yang, X., Zhang, D., Zhuang, Y.: Effective multi-modal retrieval based on stacked auto-encoders, pp. 649–660 (2014)
32. Wu, B., Yang, Q., Zheng, W.S., Wang, Y., Wang, J.: Quantized correlation hashing for fast cross-modal search. In: AAAI, pp. 3946–3952 (2015)
33. Bai, X., Yan, C., Yang, H., Bai, L., Zhou, J., Handcock, E.R.: Adaptive hash retrieval with kernel based similarity. PR **75**, 136–148 (2018)
34. Bai, X., Yang, H., Zhou, J., Ren, P., Cheng, J.: Data-dependent hashing based on p-stable distribution. TIP **23**, 5033–5046 (2014)
35. Zhen, Y., Yeung, D.Y.: Co-regularized hashing for multimodal data. In: NIPS, pp. 1376–1384 (2012)
36. Zhang, D., Li, W.J.: Large-scale supervised multimodal hashing with semantic correlation maximization. In: AAAI, pp. 2177–2183 (2014)
37. Zhu, H., Long, M., Wang, J., Cao, Y.: Deep hashing network for efficient similarity retrieval. In: AAAI, pp. 2415–2421 (2016)



# Deep Supervised Hashing with Information Loss

Xueni Zhang<sup>1</sup>(✉), Lei Zhou<sup>1</sup>, Xiao Bai<sup>1</sup>, and Edwin Hancock<sup>2</sup>

<sup>1</sup> School of Computer Science and Engineering and Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing, China

{zhangxueni, leizhou, baixiao}@buaa.edu.cn

<sup>2</sup> Department of Computer Science, University of York, York, UK  
edwin.hancock@york.ac.uk

**Abstract.** Recently, deep neural networks based hashing methods have greatly improved the image retrieval performance by simultaneously learning feature representations and binary hash functions. Most deep hashing methods utilize supervision information from semantic labels to preserve the distance similarity within local structures, however, the global distribution is ignored. We propose a novel deep supervised hashing method which aims to minimize the information loss during low-dimensional embedding process. More specifically, we use Kullback-Leibler divergences to constrain the compact codes having a similar distribution with the original images. Experimental results have shown that our method outperforms current state-of-the-art methods on benchmark datasets.

**Keywords:** Hashing · Image retrieval · KL divergence

## 1 Introduction

With the explosive growth of data in real application like image retrieval, much attention has been devoted to approximate nearest neighbor (ANN) search. Among existing ANN techniques, hashing has become one of the most popular and effective techniques due to its fast query speed and low memory cost. The crux of hashing is to embed a high dimensional vector into a set of compact binary codes while preserving the similarity of original data with Hamming distance.

Existing hashing methods can be divided into data-independent methods and data-dependent methods. Data independent methods usually choose random projections as the hash functions. The representative data-independent methods are locality sensitive hashing (LSH) [6], which directly uses random linear projections to map nearby data into similar binary codes. LSH is widely used for large scale image retrieval.

Compared with data-independent methods, data-dependent methods which try to learn hash functions from some training data can achieve comparable

or better accuracy with shorter hash codes. They can be further categorized into supervised and unsupervised methods. Retrieval of unsupervised hashing methods often rely on certain kinds of distance metric. SH [19] and ITQ [7] are two of the representative methods. In order to utilize the semantic labels of original images, many supervised hashing methods are proposed [1–3, 12, 15, 17, 21, 22].

Recently, deep learning to hash methods have shown that both feature representation and hash codes can be learned more effectively using deep neural networks, which can naturally fit any nonlinear hash functions. These deep hashing methods have created state-of-the-art results on many benchmarks. CNNH [20] is the first proposed deep hashing method, which needs two stage to learn the high-level representation and binary codes. One drawback is the hash codes cannot be updated with learned new image representation. Afterwards, deep hashing methods spring up based on different train of thought. Most deep hashing methods are supervised which utilize semantic labels to learn better binary codes. Class-label based methods aim to generate compact binary codes applicable to classification, such as DLBC [13]. Others focus on the distance between original samples. Absolute distance is used in pairwise hashing methods, such as DQN [4], DHN [25], DSH [14], DPSH [11], DSDH [10], which try to make the hamming distance between similar images as soon as possible and vice versa. While triplet methods, such as NINH [9], DSRH [24], DRSC [23], DTSH [18], consider the relative distance between images which hope to keep the hamming distance between dissimilar images farther than distance within similar images.

Although deep learning based methods have achieved great progress in image retrieval, there are some limitations of previous deep hashing methods. They mainly focus on preserving the distance relationship but ignore the information loss. We propose a novel deep hashing method based on Kullback-Leibler divergences which can constrain the compact codes having a similar distribution with the original images.

In brief, our contributions can be summarized as follows:

1. We propose a novel loss function named information loss to decrease the information loss in low-dimensional embedding process.
2. Distance similarity and distribution similarity can be simultaneously learned and mutually optimized in our deep hashing architecture.
3. Extensive experiments on three image benchmarks have shown that our method can achieve comparable performance in image retrieval applications.

## 2 Proposed Method

### 2.1 Problem Statement

Given  $N$  image samples  $X = \{x_i\}_{i=1}^N \subseteq \mathbb{R}^{d \times N}$  where each sample  $x_i$  is a  $M$ -dimensional vector, hash coding is to learn a collection of  $K$ -bit binary codes  $B \subseteq \{-1, 1\}^{K \times N}$ , where the  $i$ -th column  $b_i \subseteq \{-1, 1\}^K$  denotes the binary codes for the  $i$ -th sample  $x_i$ . The binary codes are generated by the hash function  $h(\cdot)$ ,

which can be rewritten as  $[h_1(\cdot), \dots, h_c(\cdot)]$ . For image sample  $x_i$ , its hash codes can be represented as  $b_i = h(x_i) = [h_1(\cdot), \dots, h_c(\cdot)]$ . Generally speaking, hashing is to learn a hash function to project image samples to a set of binary codes.

### 2.2 Supervised Loss

We first consider the deep hash code learning with pairwise supervised information. Usually, the label information of image datasets is given as  $Y = \{y_i\}_{i=1}^N \subseteq \mathbb{R}^{c \times N}$ , where  $y_i \subseteq \{0, 1\}^c$  corresponds to the sample  $x_i$ ,  $c$  is the number of classes. Here, the pairwise label information can be derived as:  $S = \{s_{ij}\}$ ,  $s_{ij} \subseteq \{0, 1\}$ , where  $s_{ij} = 1$  when  $x_i$  and  $x_j$  belong to the same class,  $s_{ij} = 0$  when  $x_i$  and  $x_j$  come from different classes.

Given the binary codes  $B = \{b_i\}_{i=1}^n$  for all the points, we can define the likelihood of the pairwise labels  $S = \{s_{ij}\}$  as:

$$p(s_{ij} | B) = \begin{cases} \sigma(\Omega_{ij}), & s_{ij} = 1 \\ 1 - \sigma(\Omega_{ij}), & s_{ij} = 0 \end{cases} \tag{1}$$

where  $\sigma(\Omega_{ij}) = \frac{1}{1+e^{-\Omega_{ij}}}$ , and  $\Omega_{ij} = \frac{1}{2}b_i^T b_j$ . Since there is a relationship between the hamming distance and corresponding inner product:  $dist_H(b_i, b_j) = \frac{1}{2}(K - \langle b_i, b_j \rangle)$ . We can see that the larger the inner product  $\langle b_i, b_j \rangle$  is, the smaller the corresponding  $dist_H(b_i, b_j)$  will be, and the larger  $p(1 | b_i, b_j)$  will be, which means  $b_i$  and  $b_j$  should be classified as similar, and vice versa.

By taking the negative log-likelihood of the observed pairwise labels in  $S$ , we can get the following optimization problem:

$$\min_B J_1 = -\log p(S | B) = - \sum_{s_{ij} \in S} (s_{ij}\Omega_{ij} - \log(1 + e^{\Omega_{ij}})). \tag{2}$$

It is obvious that this equation will make the hamming distance of two similar points as small as possible, and simultaneously make the hamming distance between two dissimilar points as large as possible, which is exactly the goal of supervised hashing with pairwise labels.

Although pairwise label supervision can preserve the distance similarity between original images, the label information is not fully exploited. It is a reasonable assumption that good binary codes should contain enough semantic information to preserve semantic similarity between images. In other words, the learned binary codes should be ideal for classification.

Consider the binary codes learning problem in the linear classification framework, the multi-class classification problem can be represented as the following formulation:

$$y = W^T b = [W_1^T b, \dots, W_C^T b]^T \tag{3}$$

where  $w_k \in \mathbb{R}^{L \times 1}$ ,  $k = 1, \dots, C$  is the classification vector for class  $k$  and  $y \in \mathbb{R}^{L \times 1}$  is the label vector, of which the maximum item indicates the assigned class of  $x$ . Thus, we can obtain the following optimization problem:

$$\min_{B, W} J_2 = \sum_{i=1}^n L(y_i, W^T b_i) + \lambda \|W\|^2 \tag{4}$$

where  $\lambda$  is the regularization parameter;  $y_i \in \mathfrak{R}^{C \times 1}$  is the ground truth label of  $x_i$ , where  $y_{ki} = 1$  if  $x_i$  belongs to class  $k$  and  $y_{ki} = 0$  if don't.  $\|\cdot\|$  is the  $\ell_2$  norm for vectors and Frobenius norm for matrices.  $L(\cdot)$  is the loss function for classification. The problem can be rewritten as

$$\min_{B, W} J_2 = \sum_{i=1}^n \|y_i - W^T b_i\|^2 + \lambda \|W\|^2 \quad (5)$$

### 2.3 Information Loss

Preserving distance and semantic similarity is an important part of hashing method. However, existing methods just take into account the relationship of one point or point-pairs. Considering good embedding needs to keep not only local structure but also global distribution, we introduce Kullback-Leibler divergence to constrain the low-dimensional distribution.

First, we construct conditional probabilities from Euclidean distance to represent similarities between data points. The similarity of  $x_i$  to  $x_j$  is the conditional probability,  $p_{j|i}$ , that  $x_i$  would pick  $x_j$  as its neighbor if neighbors were picked in proportion to their probability density under a Gaussian centered at  $x_i$ . For nearby datapoints,  $p_{j|i}$  is relatively high, whereas for widely separated datapoints,  $p_{j|i}$  will be almost infinitesimal. We can see that, this similarity quite matches the essence of retrieval. The conditional probability can be defined as

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)} \quad (6)$$

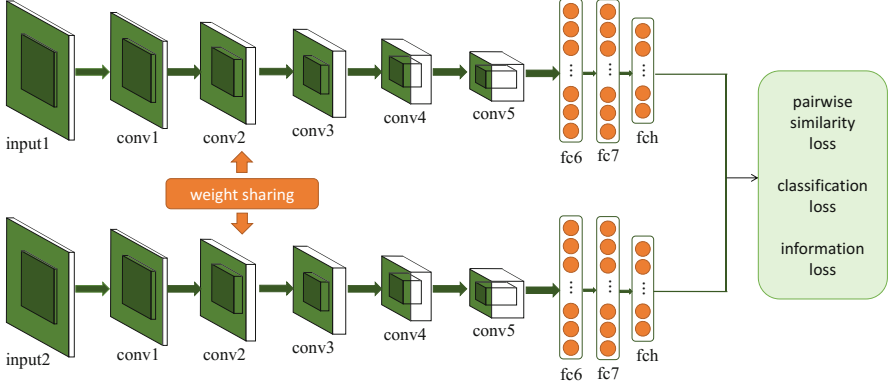
Furthermore, the joint probability can be derived as  $p_{ij} = \frac{p_{i|j} + p_{j|i}}{2^n}$ .

Following t-SNE [16], to alleviate the crowding problem, we use a probability distribution that has much heavier tails than a Gaussian to convert distances into probabilities in the low-dimensional space. Specifically, we employ a Student t-distribution with one degree of freedom (which is the same as a Cauchy distribution) as the heavy-tailed distribution. The joint probabilities  $q_{ij}$  are defined as

$$q_{ij} = \frac{(1 + \|b_i - b_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|b_k - b_l\|^2)^{-1}} \quad (7)$$

If the binary points  $b_i$  and  $b_j$  correctly model the similarity between the high-dimensional datapoints  $x_i$  and  $x_j$ , the joint probabilities  $p_{ij}$  and  $q_{ij}$  will be equal. Therefore, our goal is to find a low-dimensional binary representation that minimizes the mismatch between  $p_{ij}$  and  $q_{ij}$ . This can be measured by Kullback-Leiber divergence with which  $q_{ij}$  models  $p_{ij}$ . The information loss can be represented as follows:

$$J_3 = \sum KL(P_i \| Q_i) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (8)$$



**Fig. 1.** The architecture of our proposed method.

To sum up, the total loss function can be achieved by combining pairwise similarity loss, classification loss and information loss:

$$J = J_1 + \alpha J_2 + \beta J_3 \quad (9)$$

## 2.4 Optimization

In order to have a fair comparison with previous deep hashing methods, we also choose the CNN-F network architecture to learn the feature representation and hash function. Since using pairwise-label supervision, our model consists of two separate CNNs which share the same weights. Each CNN includes 5 convolutional layers and 2 fully connected layers. The pipeline is shown in Fig. 1.

Obviously, the minimization of the obtained loss function in Sect. 2.3 is a discrete optimization problem, which is hard to optimize directly. We solve this problem by introducing an auxiliary variable, the output of the last fully connected layer,  $u_i$  and make  $b_i = \text{sgn}(u_i)$ . It can be represented as:

$$u_i = M^T \phi(x_i; \theta) + v \quad (10)$$

where  $\theta$  denotes all the parameters of the previous layers,  $\phi(x_i; \theta)$  denotes the output of the penultimate fully connected layer,  $M$  represents the weight matrix, and  $v$  is the bias term. Then we can reformulate the optimization problem as the following equivalent one:

$$\begin{aligned} \min J' = & - \sum_{s_{ij} \in S} (s_{ij} \Psi_{ij} - \log(1 + e^{\Psi_{ij}})) + \alpha \sum_{i=1}^n \|y_i - W^T u_i\|^2 \\ & + \lambda \|W\|^2 + \beta \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} + \eta \sum_{i=1}^n \|b_i - u_i\|^2 \end{aligned} \quad (11)$$



where  $\Psi_{ij} = \frac{1}{2}u_i^T u_j$ ,  $q_{ij} = \frac{(1+\|u_i-u_j\|^2)^{-1}}{\sum_{k \neq l} (1+\|u_k-u_l\|^2)^{-1}}$ .

In our method, we use an alternating strategy to learn these parameters. In other words, we optimize one parameter with other parameters fixed. Firstly, the  $b_i$  can be directly optimized by

$$b_i = \text{sgn}(u_i) = \text{sgn}(M^T \phi(x_i; \theta) + v) \quad (12)$$

For the other parameters, we use back-propagation(BP) algorithm for learning. In particular, we can compute the derivatives of the loss function with respect to  $u_i$  as follows:

$$\begin{aligned} \frac{\partial J}{\partial u_i} = & \frac{1}{2} \sum_{j: s_{ij} \in S} (a_{ij} - s_{ij})u_j + \frac{1}{2} \sum_{j: s_{ij} \in S} (a_{ji} - s_{ji})u_j + 2\eta(u_i - b_i) - 2\alpha W^T \\ & (y_i - W^T u_i) - 2\beta \sum_i (1 + \|z_i - u_j\|^2)^{-1} \times (p_{ij} - q_{ij})(z_i - u_j) \end{aligned}$$

where  $a_{ij} = \sigma(\frac{1}{2}u_i^T u_j)$ . Then, we can update the other parameters by back propagation:

$$\begin{aligned} \frac{\partial J}{\partial M} &= \phi(x_i; \theta) \left( \frac{\partial J}{\partial u_i} \right)^T, \quad \frac{\partial J}{\partial v} = \frac{\partial J}{\partial u_i}, \quad \frac{\partial J}{\partial \phi(x_i; \theta)} = M \frac{\partial J}{\partial u_i}, \\ \frac{\partial J}{\partial W} &= -2 \sum_{i=1}^n u_i (y_i - W^T u_i) + 2\lambda W, \\ \frac{\partial J}{\partial z_i} &= 2 \sum_j (1 + \|z_i - u_j\|^2)^{-1} \times (p_{ij} - q_{ij})(z_i - u_j) + M \frac{\partial J}{\partial u_i} \frac{\partial \phi(x_i; \theta)}{\partial z_i} \end{aligned}$$

### 3 Experiments

#### 3.1 Datasets and Evaluation Criterion

We conduct experiments on two widely used benchmark datasets, CIFAR-10 [8] and NUS-WIDE [5]. The CIFAR-10 dataset contains 60,000 color images of size  $32 * 32$ , which are categorized into 10 classes and 6,000 images for each class. Each image is only associated with one class. The NUS-WIDE dataset contains nearly 27,000 color images from the web. Different from CIFAR-10, NUS-WIDE is a multi-label dataset in which each image is annotated with one or multiple class labels in 81 semantic concepts. Following the setting in [10, 11, 20, 23], we use a subset of 195,834 images which are annotated with 21 most frequent classes. For each of the 21 classes, at least 5,000 images are annotated with it.

We employ mean average precision (MAP) to evaluate the performance of our method and baselines similar to most previous work. For these datasets, the similar pairs are constructed according to the image labels: two images will be considered similar only if they share at least one common semantic label.

### 3.2 Baselines and Setting

We compare our method with several state-of-the-art hashing methods. They can be roughly divided into two groups: traditional hashing methods and deep hashing methods, while the traditional methods can be further divided into unsupervised and supervised methods. Unsupervised hashing methods include SH [19], ITQ [7]. Supervised methods include KSH [15], FastH [12], LFH [22], and SDH [17]. Both the hand-crafted features and the features extracted by CNN-F network architecture are used as the input for the traditional hashing methods. Similar to previous works, when using handcrafted features, we use a 512-dimensional GIST descriptor to represent images of CIFAR-10 dataset, and a 1134-dimensional feature vector to represent images of NUS-WIDE dataset, which is the concatenation of a 64-D color histogram, a 144-D color correlogram, a 73-D edge direction histogram, a 128-D wavelet texture, a 225-D block-wise color moments and a 500-D BoW representation based on SIFT descriptors.

The deep hashing methods include CNNH [20], NINH [9], DSRH [24], DSCH [23], DRSCH [23], DQN [4], DHN [25], DPSH [11], DTSH [18], DSDH [10]. Although DPSH, DTSH and DSDH are based on the CNN-F network architecture and DQN, DHN, DSRH are based on AlexNet architecture, both the CNN-F and AlexNet architectures consist of five convolutional layers and two fully connected layers. So they are still comparable. In order to have a fair comparison, most of the results are directly reported from previous works.

We compare our method to baselines under the following two kinds of experimental settings. For the first setting, we randomly select 100 images per class (1,000 images in total) as the test query set, 500 images per class (5,000 images in total) as the training set in CIFAR-10. For NUS-WIDE dataset, we randomly sample 100 images per class (2,100 images in total) as the test query set, 500 images per class (10,500 images in total) as the training set. As for the second experimental setting, in CIFAR-10, 1,000 images per class are selected as the test query set, the remaining 50,000 images are used as the training set. In NUS-WIDE, 100 images per class are randomly sampled as the test query images, the remaining 193,734 images are used as the training set. Since NUS-WIDE contains a huge number of images, when computing MAP for NUS-WIDE, we only consider the top 5,000 returned neighbors under the first setting and top 50,000 under the second experimental setting.

### 3.3 Performance Evaluation

**Results Under the First Experimental Setting.** The MAP results of all methods on CIFAR-10 and NUS-WIDE under the first experimental setting are listed in Table 1. We can see that on CIFAR-10 dataset, the MAP result of our method is more than twice as much as SDH, FastH and ITQ, which are the best several kinds of traditional hashing methods. For the deep hashing methods, our proposed method which consider both supervised information and distribution similarity, nearly improves the performance of DSDH by 2%. These results verify that the proposed information loss is benefit to obtain good binary codes. From

**Table 1.** Mean Average Precision (MAP) under the first experimental setting. The best performance is shown in boldface.

Method	CIFAR-10				NUS-WIDE			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
SH	0.127	0.128	0.126	0.129	0.454	0.406	0.405	0.400
ITQ	0.162	0.169	0.172	0.175	0.452	0.468	0.472	0.477
LFH	0.176	0.231	0.211	0.253	0.571	0.568	0.568	0.585
KSH	0.303	0.337	0.346	0.356	0.556	0.572	0.581	0.588
SDH	0.285	0.329	0.341	0.356	0.568	0.600	0.608	0.637
FastH	0.305	0.349	0.369	0.384	0.621	0.650	0.665	0.687
CNNH	0.439	0.511	0.509	0.522	0.611	0.618	0.625	0.608
NINH	0.552	0.566	0.558	0.581	0.674	0.697	0.713	0.715
DHN	0.555	0.594	0.603	0.621	0.708	0.735	0.748	0.758
DQN	0.554	0.558	0.564	0.580	0.768	0.776	0.783	0.792
DPSH	0.713	0.727	0.744	0.757	0.752	0.790	0.794	0.812
DTSH	0.710	0.750	0.765	0.774	0.773	0.808	0.812	0.824
DSDH	<b>0.740</b>	0.786	0.801	0.820	0.776	0.808	0.820	0.829
<b>Ours</b>	0.738	<b>0.792</b>	<b>0.822</b>	<b>0.841</b>	<b>0.781</b>	<b>0.823</b>	<b>0.837</b>	<b>0.840</b>

**Table 2.** Mean Average Precision (MAP) under the second experimental setting. The best performance is shown in boldface.

Method	CIFAR-10				NUS-WIDE			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
DSRH	0.608	0.611	0.617	0.618	0.609	0.618	0.621	0.631
DSCH	0.609	0.613	0.617	0.620	0.592	0.597	0.611	0.609
DRSCH	0.615	0.622	0.629	0.631	0.618	0.622	0.623	0.628
DPSH	0.763	0.781	0.795	0.807	0.715	0.722	0.736	0.741
DTSH	0.915	0.923	0.925	0.926	0.756	0.776	0.785	0.799
DSDH	0.935	0.940	0.939	0.939	0.815	0.814	0.820	0.821
<b>Ours</b>	<b>0.941</b>	<b>0.945</b>	<b>0.948</b>	<b>0.952</b>	<b>0.843</b>	<b>0.849</b>	<b>0.857</b>	<b>0.862</b>

Table 1, it is also shown that our method outperforms the state-of-the-art on the NUS-WIDE dataset.

**Results Under the Second Experimental Setting.** We also compare these hashing methods under the second experimental setting, which contains more training images. Table 2 lists MAP results for different methods, from which we can see that almost all deep hashing methods perform better than under the first setting. It means that they are more suitable for large-scale datasets.

With sufficient training and adequate guidance by loss function, our method outperforms the baseline works.

**Comparison to Traditional Hashing Methods Using Deep Features.** To further verify the effective of our loss, we compare our method with traditional hashing methods which use deep features extracted by CNN-F pretrained on ImageNet. The results are reported in Table 3. We can see that all traditional hashing methods have a great performance improvement with CNN features. Particularly, the performance of FastH with CNN features on CIFAR-10 is nearly twice than that of hand-crafted features. However, there is still great gap between our method and traditional methods.

**Table 3.** Mean Average Precision (MAP) under the first experimental setting. The best performance is shown in boldface.

Method	CIFAR-10				NUS-WIDE			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
SH+CNN	0.183	0.164	0.161	0.161	0.621	0.616	0.615	0.612
ITQ+CNN	0.237	0.246	0.255	0.261	0.719	0.739	0.747	0.756
LFH+CNN	0.208	0.242	0.266	0.339	0.695	0.734	0.739	0.759
KSH+CNN	0.488	0.539	0.548	0.563	0.768	0.786	0.790	0.799
SDH+CNN	0.478	0.557	0.584	0.592	0.780	0.804	0.815	0.824
FastH+CNN	0.553	0.607	0.619	0.636	0.779	0.807	0.816	0.825
<b>Ours</b>	<b>0.738</b>	<b>0.792</b>	<b>0.822</b>	<b>0.841</b>	<b>0.781</b>	<b>0.823</b>	<b>0.837</b>	<b>0.840</b>

## 4 Conclusion

In this paper, we proposed a novel deep hashing method. In addition to use the pairwise label information and the classification information, we also introduced the KL divergence to constrain the information loss during the low-dimensional embedding, which can preserve both local and global structures. Extensive experiments show that our method can achieve comparable performance in image retrieval applications.

**Acknowledgement.** This work was supported by the National Natural Science Foundation of China project no. 61772057, in part by Beijing Natural Science Foundation project no. 4162037, and the support funding from State Key Lab. of Software Development Environment.

## References

1. Bai, X., Yan, C., Ren, P., Bai, L., Zhou, J.: Discriminative sparse neighbor coding. *Multimed. Tools Appl.* **75**(7), 4013–4037 (2016)
2. Bai, X., Yan, C., Yang, H., Bai, L., Zhou, J., Hancock, E.R.: Adaptive hash retrieval with kernel based similarity. *Pattern Recognit.* **75**, 136–148 (2018)
3. Bai, X., Yang, H., Zhou, J., Ren, P., Cheng, J.: Data-dependent hashing based on p-stable distribution. *IEEE Trans. Image Process.* **23**(12), 5033–5046 (2014)
4. Cao, Y., Long, M., Wang, J., Zhu, H., Wen, Q.: Deep quantization network for efficient image retrieval. In: *AAAI*, pp. 3457–3463 (2016)
5. Chua, T.S., Tang, J., Hong, R., Li, H., Luo, Z., Zheng, Y.: NUS-WIDE: a real-world web image database from National University of Singapore. In: *Proceedings of the ACM International Conference on Image and Video Retrieval*, p. 48. ACM (2009)
6. Gionis, A., Indyk, P., Motwani, R., et al.: Similarity search in high dimensions via hashing. *VLDB* **99**, 518–529 (1999)
7. Gong, Y., Lazebnik, S., Gordo, A., Perronnin, F.: Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(12), 2916–2929 (2013)
8. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images (2009)
9. Lai, H., Pan, Y., Liu, Y., Yan, S.: Simultaneous feature learning and hash coding with deep neural networks. arXiv preprint [arXiv:1504.03410](https://arxiv.org/abs/1504.03410) (2015)
10. Li, Q., Sun, Z., He, R., Tan, T.: Deep supervised discrete hashing. In: *Advances in Neural Information Processing Systems*, pp. 2479–2488 (2017)
11. Li, W.J., Wang, S., Kang, W.C.: Feature learning based deep supervised hashing with pairwise labels. arXiv preprint [arXiv:1511.03855](https://arxiv.org/abs/1511.03855) (2015)
12. Lin, G., Shen, C., Shi, Q., Van den Hengel, A., Suter, D.: Fast supervised hashing with decision trees for high-dimensional data. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1971–1978. IEEE (2014)
13. Lin, K., Yang, H.F., Hsiao, J.H., Chen, C.S.: Deep learning of binary hash codes for fast image retrieval. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 27–35. IEEE (2015)
14. Liu, H., Wang, R., Shan, S., Chen, X.: Deep supervised hashing for fast image retrieval. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2064–2072 (2016)
15. Liu, W., Wang, J., Ji, R., Jiang, Y.G., Chang, S.F.: Supervised hashing with kernels. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2074–2081. IEEE (2012)
16. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(Nov), 2579–2605 (2008)
17. Shen, F., Shen, C., Liu, W., Shen, H.T.: Supervised discrete hashing. In: *CVPR*, vol. 2, p. 5 (2015)
18. Wang, X., Shi, Y., Kitani, K.M.: Deep supervised hashing with triplet labels. In: *Lai, S.H., Lepetit, V., Nishino, K., Sato, Y. (eds.) ACCV 2016. LNCS*, vol. 10111, pp. 70–84. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-54181-5\\_5](https://doi.org/10.1007/978-3-319-54181-5_5)
19. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: *Advances in Neural Information Processing Systems*, pp. 1753–1760 (2009)
20. Xia, R., Pan, Y., Lai, H., Liu, C., Yan, S.: Supervised hashing for image retrieval via image representation learning. In: *AAAI*, vol. 1, p. 2 (2014)

21. Yang, H., et al.: Maximum margin hashing with supervised information. *Multimed. Tools Appl.* **75**(7), 3955–3971 (2016)
22. Zhang, P., Zhang, W., Li, W.J., Guo, M.: Supervised hashing with latent factor models. In: *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 173–182. ACM (2014)
23. Zhang, R., Lin, L., Zhang, R., Zuo, W., Zhang, L.: Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. *IEEE Trans. Image Process.* **24**(12), 4766–4779 (2015)
24. Zhao, F., Huang, Y., Wang, L., Tan, T.: Deep semantic ranking based hashing for multi-label image retrieval. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1556–1564. IEEE (2015)
25. Zhu, H., Long, M., Wang, J., Cao, Y.: Deep hashing network for efficient similarity retrieval. In: *AAAI*, pp. 2415–2421 (2016)



# Single Image Super Resolution via Neighbor Reconstruction

Zhihong Zhang<sup>1</sup>, Zhuobin Xu<sup>1</sup>, Zhiling Ye<sup>1</sup>, Yiquan Hu<sup>2(✉)</sup>, Lixin Cui<sup>3</sup>,  
and Lu Bai<sup>3</sup>

<sup>1</sup> Xiamen University, Xiamen, Fujian, China

<sup>2</sup> Zhongshan Hospital affiliated with Xiamen University, Xiamen, China  
hyq0826@yahoo.com

<sup>3</sup> Central University of Finance and Economics, Beijing, China

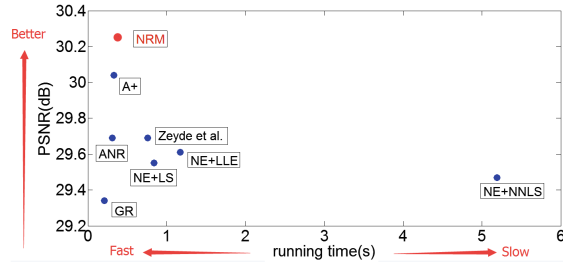
**Abstract.** Super Resolution (SR) is a complex, ill-posed problem where the aim is to construct the mapping between the low and high resolution manifolds of image patches. Anchored neighborhood regression for SR (namely A+ [15]) has shown promising results. In this paper we present a new regression-based SR algorithm that overcomes the limitations of A+ and benefits from an innovative and simple Neighbor Reconstruction Method (NRM). This is achieved by vector operations on an anchored point and its corresponding neighborhood. NRM reconstructs new patches which are closer to the anchor point in the manifold space. Our method is robust to NRM sparsely-sampled points: increasing PSNR by 0.5 dB compared to the next best method. We comprehensively validate our technique on standardised datasets and compare favourably with the state-of-the-art methods: we obtain PSNR improvement of up to 0.21 dB compared to previously-reported work.

**Keywords:** Super resolution · Manifold learning  
Neighbor reconstruction

## 1 Introduction

The purpose of single image super-resolution (SR) is to estimate a high resolution (HR) image from a single low resolution (LR) image. It provides a way to enhance the existing images which were generated by delayed imaging equipment or limited imaging conditions, and have been widely studied in recent years. Acquiring a HR estimation from an LR observation is an ill-posed problem and so priors of high quality images are normally relied on in the estimation process. Based on the different priors, existing single image SR methods can be broadly classified into three categories: interpolation-based methods [6, 7], reconstruction-based methods [1, 17] and example learning-based methods [2–5, 8, 14, 15, 18].

**Electronic supplementary material** The online version of this chapter ([https://doi.org/10.1007/978-3-319-97785-0\\_39](https://doi.org/10.1007/978-3-319-97785-0_39)) contains supplementary material, which is available to authorized users.



**Fig. 1.** Average PSNR (dB) vs time (s) of our algorithm (NRM) compared to other SR methods. We largely improve (red) over the original example based single image super-resolution methods (blue), i.e. our NRM method is 0.21 dB better than A+ [15] and 0.91 dB better than the Global Regression (GR) [14]. Results reported on Set5 with magnification 4. (Color figure online)

Among the above mapping-based methods, neighbor embedding approaches have achieved great research interests. In [14], Timofte *et al.* proposed a highly efficient and effective SR algorithm called ANR, which maps the LR patches onto the HR domain using the projections learned from neighborhoods. Specifically, it relaxes the  $\ell_1$ -norm regularization commonly used in most of the neighbor embedding and sparse coding approaches [16, 17] to a  $\ell_2$ -norm regularized regression which can be solved offline and stored for each dictionary atom/anchor. This results in large speed benefits. Subsequently, those authors proposed an improved variant of the ANR method called A+ [15] that learns the regressors from the locally nearest training LR and HR patches instead of the small dictionary. It thus better utilizes the prior data to achieve improved performance. Under the framework of A+, many notable methods such as the Half Hypersphere Confinement Regression (HHCR) [11], the Patch Symmetry Collapse (PSyCo) [9] and RFL [12] were proposed.

Although the A+ method [15] has achieved great success in delivering high quality HR estimation, it has two serious limitations: First, to obtain dense sample patches, A+ needs to harvest data images with different scales repeatedly, resulting in a large amount of computation and storage; Second, even if A+ does a so-called densely harvesting, we find that these patches are still too sparse for the high dimension space.

## 1.1 Contributions

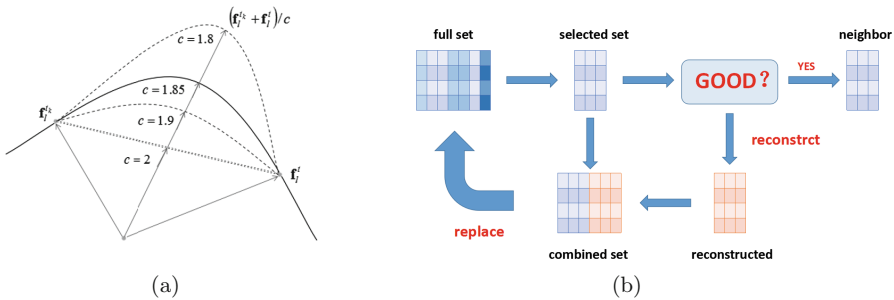
In this paper, we propose a novel and simple neighbor reconstruction method and extend the concept of A+ resulting in a significant improvement.

1. Compared with A+, our method utilizes fewer features to construct a closer neighbor and that results in a more accurate reconstruction coefficient vector  $\mathbf{x}$ . Specifically, we present a new neighbor reconstruction method which adds an anchor point and its corresponding neighbor features together and divides



the result by a scalar to generate a much closer neighbor. Compared with the A+ method, our method requires fewer features to generate a closer neighbor set.

2. Meanwhile, we have also designed a new projector which has much better numerical stability to adapt to our new problem. As in A+, to obtain the low resolution reconstruction coefficient vector  $\mathbf{x}$ , we solve a regularized and over-completed least-squares problem detailed in Eq. (4). We present a numerically stable projector Eq. (6) to supplement our method.
3. In this case, by benefiting from closer neighbor we obtain a more accurate reconstruction coefficient vector  $\mathbf{x}$  leading to an improvement circa 0.1–0.21 dB over A+. Moreover, with fixed memory, more anchor points can be trained leading to much better generalization. Figure 1 shows improved quantitative performance.



**Fig. 2.** Illustration of sample reconstruction. (a) geometric interpretation of neighborhood reconstruction. The figure shows how to create a cosine similarity closer point  $(\mathbf{f}_i^{t_k} + \mathbf{f}_i^t)/c$  by using  $\mathbf{f}_i^t$  and its neighbor  $\mathbf{f}_i^{t_k}$ .  $c$  is an adjustable parameter to make  $(\mathbf{f}_i^{t_k} + \mathbf{f}_i^t)/c$  be close to the intrinsic manifold, namely the solid line. In this figure, when  $c = 1.85$ ,  $(\mathbf{f}_i^{t_k} + \mathbf{f}_i^t)/c$  can fall on the intrinsic manifold. (b) shows how to do neighbor reconstruction process iteratively.

## 2 Analysis of Manifold-Based Single Image SR

We analyse in more detail the A+ technique and explain the limitations of their method. All of our analysis is based on a basic property of the manifold: if an assigned neighbor is close enough then the local manifold subspace can be well described by the observed coordinates of the neighbor. Namely, if the neighbor of aimed anchor point is close enough, we can use our coordinated points to describe the inherent property of the manifold. The well-known Local Linear Embedding (LLE) [10] was proposed based on this property and A+ method was, in turn, motivated by LLE. There are two major deficiencies of A+ method.

1. To harvest dense sample patches, the A+ method samples patches at different scales. If we generate dense patches with the A+ method on a large database,

it is massively expensive in both computation and memory. For example, for a 91-image dataset, to obtain dense patches around the anchored point, A+ method attempts to harvest 12 times at different scales resulting in about 5 millions patches.

2. A simple estimation shows that the patches harvested with the A+ method are not close enough. In practice the dimension of features drawn from the low dimensional patches is around 30. We aim to find a neighbor which lies within an anchor point centred hypersphere whose radius is 0.1. Without loss of generality, supposing that features are normalized and uniformly distributed, at least  $10^{30}$  features are needed to reconstruct that required neighbor while only 5 million features are used in A+.

## 2.1 A Manifold-Based Model

We analyse the generalisation capacity of manifold-based single image SR. Firstly, some notation is introduced. Suppose  $\mathbf{p}_h$  are small sampled patches which are directly cropped from raw training images.  $\mathbf{p}_l$  is downsampled patches from  $\mathbf{p}_h$ . And that  $\mathbf{f}_l$  and  $\mathbf{f}_h$  are normalized features extracted from  $\mathbf{p}_l$  and  $\mathbf{p}_h$  respectively by feature extractors,  $\mathbf{f}_l = K_l(\mathbf{p}_l)$ ,  $NNNNNNNNNNNnn\mathbf{f}_h = K_h(\mathbf{p}_h)$ , where  $K_l$  and  $K_h$  is linear feature extractors.

Further suppose that  $\widehat{M}_l$  and  $\widehat{M}_h$  are sampled manifolds corresponding to low-dimensional and high-dimensional feature spaces, namely,  $\widehat{M}_l = \{\mathbf{f}_l^{(i)}\}_{i=1}^n$ ,  $\widehat{M}_h = \{\mathbf{f}_h^{(i)}\}_{i=1}^n$ , where  $n$  is the number of extracted features in the low-dimensional or high-dimensional feature space. Suppose  $M_l$  and  $M_h$  are continuous ground truth manifolds corresponding to the LR and HR feature spaces. These two manifolds are structurally similar at local subspace. The relationship between the sampled manifolds and ground truth manifolds is:  $M_l = \lim_{n \rightarrow \infty} \widehat{M}_l$ ,  $M_h = \lim_{n \rightarrow \infty} \widehat{M}_h$ .

There is an important one-to-one mapping,  $H(\mathbf{p}_h) = \mathbf{f}_l(\in \widehat{M}_l)$ , which is a naturally formed result when we are preparing the low and high patches. In practice we firstly train an LR dictionary  $\mathbf{D}_l$ ,

$$\mathbf{D}_l, \alpha^i = \arg \min_{\mathbf{D}_l, \alpha^i} \sum_i \|\mathbf{f}_l^{(i)} - \mathbf{D}_l \alpha^i\|_2^2 + \lambda^2 \|\alpha^i\|_2^2. \quad (1)$$

Each column of  $\widehat{D}_l$  is called as an atom,  $\mathbf{d}_l$ . In A+ researchers use atoms as anchor points in  $\widehat{M}_l$  to anchor offline projectors. Given a target low dimensional feature  $\mathbf{f}_l^t$  researchers use a neighbor set of its nearest atom to reconstruct  $\mathbf{f}_l^t$ . This reconstruction leads to a reconstruction parameter  $\mathbf{x}$ . The reconstruction process can be formulated as,

$$\mathbf{x} = \arg \min_{\mathbf{x}} \|\mathbf{f}_l^t - \mathbf{N}_l(\mathbf{d}_l)\mathbf{x}\|_2^2 + \lambda^2 \|\mathbf{x}\|_2^2 \quad (2)$$

where  $\mathbf{N}_l(\mathbf{d}_l)$  is a neighbor set of  $\mathbf{d}_l$ . The Eq. (2) can be solved with a closed-form,

$$\mathbf{x} = \mathbf{P}\mathbf{f}_l^t,$$

where  $\mathbf{P} = (\mathbf{N}_l^T \mathbf{N}_l + \lambda^2 \mathbf{I})^{-1} \mathbf{N}_l^T$ . Obviously for each atom its corresponding  $\mathbf{P}$  can be prepared offline. With parameter  $\mathbf{x}$  and the one-to-one mapping  $H(\mathbf{p}_h) = \mathbf{f}_l(\in \widehat{M}_l)$  high-dimensional patch  $\mathbf{p}_l$  can be reconstructed in the way used in LLE [10].

The SR problem in the NE framework is to construct a generalized function  $G(\mathbf{f}_l) \approx \mathbf{p}_h : M_l \rightarrow P_h$  where  $P_h$  is continuous high-dimensional image patches manifold space. Referring to the former one-to-one mapping  $H$ . During testing, a given evaluation criterion is used, such as PSNR (Peak Signal to Noise Ratio), SSIM (Structural Similarity Index) and IFC (Information Fidelity Criterion), to estimate the performance of  $G$ . The estimator is,

$$C(I(G(\mathbf{f}_l^{(i)})) - I(\mathbf{p}_h^{(i)})),$$

where  $C$  is a chosen image evaluation criterion,  $I$  is a patch combining function which generates final patch-combining images. And  $\mathbf{f}_l^{(i)} \in \widehat{M}_l, \mathbf{p}_h^{(i)} \in \widehat{P}_h, \widehat{P}_h$  are HR patch sets harvested from the training database.

The object fun of SR is,

$$\max_G \sum_i C(I(G(\mathbf{f}_l^{(i)})) - I(\mathbf{p}_h^{(i)})).$$

## 2.2 The Neighbor Reconstruction Method

As in A+ when we are training the function  $G$ , given a target feature  $\mathbf{f}_l^t$ , we want to obtain a reconstruction coefficient vector  $\mathbf{x}$ . Then we directly transfer the coefficient vector into HR patch space, and construct the interest  $\mathbf{p}_h^t$  with one-to-one mapping  $H$ . In the HR patch space we use the coefficient vector  $\mathbf{x}$  and the corresponding neighbor to reconstruct target  $\mathbf{p}_h^t$ . So it is crucial to choose a good neighbor. Inspired by a Euclidean theorem in plane space, namely the parallelogram axiom of vectors, we have designed a neighbor reconstruction method denoted NRM, more detailed in Fig. 2(a). Based on the cosine similarity metric we construct a closer, or more highly correlative, neighbor set for  $\mathbf{f}_l^t$  which will be beneficial in generating a more accurate reconstruction coefficient  $\mathbf{x}$ .

Denote the neighbors  $\mathbf{N}_l(\mathbf{d}_l)$  of  $\mathbf{f}_l^t$  as the set of vectors  $[\mathbf{f}_l^{t_1}, \mathbf{f}_l^{t_2}, \dots, \mathbf{f}_l^{t_k}]$ . We concatenate the central point and its corresponding neighbors together as column in the matrix  $\bar{\mathbf{F}} = [\mathbf{f}_l^{t_1}, \mathbf{f}_l^{t_2}, \dots, \mathbf{f}_l^{t_k}, \mathbf{f}_l^t]$ . We induce a reconstruction operator,

$$\mathbf{R} = \begin{bmatrix} \frac{1}{c} & 0 & \dots & 0 & 0 \\ 0 & \frac{1}{c} & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \frac{1}{c} & 0 \\ \frac{1}{c} & \frac{1}{c} & \frac{1}{c} & \frac{c}{c} & 1 \end{bmatrix} \in \mathbb{R}^{(k+1) \times (k+1)} \tag{3}$$

where  $c(>1)$  is an adjustable parameter. For the  $j$ th ( $1 \leq j < k + 1$ ) column  $R_j$ , it can generate the  $j$ th reconstructed neighbor  $\frac{1}{c} \mathbf{f}_l^t + \frac{1}{c} \mathbf{f}_l^{t_j}$  by the right multiplication  $\bar{\mathbf{F}} R_j$ . For the  $(k + 1)$ th column, it is used to preserve central point  $\mathbf{f}_l^t$  for the next iteration. In NRM, reconstruction manipulation is achieved in parallel

by right multiplying  $\mathbf{R}$  by  $\bar{\mathbf{F}}$ . This manipulation can be done achieved iteratively.  $\bar{\mathbf{F}}^{(r)} = \bar{\mathbf{F}}\mathbf{R}^r$  ( $r \in \{0, 1, 2, 3, \dots, s\}$ ) where  $s$  is a truncation number. After operating on  $\bar{\mathbf{F}}$  for  $s$  times, NRM collects  $\pm\bar{\mathbf{F}}^{(r)}$  as a large set  $\mathfrak{F} = \{\pm\bar{\mathbf{F}}^{(r)}\}_{r=0}^s$ . The final step in NRM is to select  $k$  the nearest points for  $\mathbf{f}_l^t$  from  $\mathfrak{F}$  to replace the original neighbor set. Further details of the iterative approach are shown in Fig. 2(b).

$-1$  before  $\bar{\mathbf{F}}^{(r)}$  reverse the sign, if we want to employ the parallelogram axiom of vectors to efficiently generate a closer neighbor feature, we must ensure  $\mathbf{f}_l^t$  and  $\mathbf{f}_l^{t_j}$  lie on the same side of the anchor. Considering the existence of antipodal points we reverse the neighbor set by multiplying a negative one ( $-1$ ) on its features, and utilize these reversed antipodal points to generate reconstructed points.

### 2.3 Solving the Model

First, given a target feature  $\mathbf{f}_l^t$ , we employ NRM to generate a corresponding neighbor set  $\mathbf{N}_l$ . To obtain reconstruction coefficients  $\mathbf{x}$  in a low resolution space, we need to solve the optimization problem,

$$\min_{\mathbf{x}} \|\mathbf{f}_l^t - \mathbf{N}_l\mathbf{x}\|_2^2 + \lambda^2 \|\mathbf{x}\|_2^2. \quad (4)$$

For the problem, in A+, the solution is,

$$\mathbf{x} = \mathbf{P}\mathbf{f}_l^t,$$

where the projector  $\mathbf{P} = (\mathbf{N}_l^T\mathbf{N}_l + \lambda^2\mathbf{I})^{-1}\mathbf{N}_l^T$ .

In our method, we reconstruct a closer neighbor leading to a greater condition number of  $\mathbf{N}_l$ . If we still apply the projector  $\mathbf{P}$  which is deduced with normal equation method to obtain  $\mathbf{x}$  in Eq. (4), this will lead to poor results. Because in normal equation method an inverse of matrix is needed to be computed, a large condition number will lead to a big numerical error which can be a deviation from our best results about 6 dB as shown in Fig. 3.

To regular this great condition number problem we design a new projector based on matrix QR decomposition in which we do not have to compute a inverse of matrix. Rewriting Eq. (4) in the least-squares form:

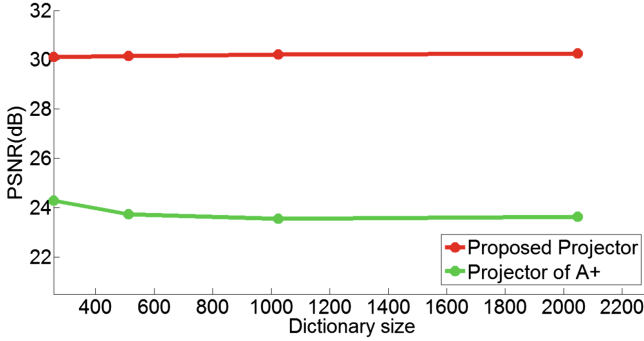
$$\min \left\| \begin{bmatrix} \lambda\mathbf{I} \\ \mathbf{N}_l \end{bmatrix}_{(m+n,n)} \mathbf{x} - \begin{bmatrix} \mathbf{O} \\ \mathbf{f}_l^t \end{bmatrix}_{(m+n,1)} \right\|_2^2, \quad (5)$$

where  $m$  is the dimension of the features in  $\mathbf{N}_l$ ,  $n$  is the number of neighbor features, ( $m \ll n$ ). And  $\mathbf{N}_l \in \mathbb{R}^{m \times n}$ ,  $\lambda\mathbf{I} \in \mathbb{R}^{n \times n}$ ,  $\mathbf{O} \in \mathbb{R}^{n \times 1}$ ,  $\mathbf{f}_l^t \in \mathbb{R}^{m \times 1}$ .

Applying the QR decomposition method to Eq. (5) gives:

$$\begin{bmatrix} \lambda\mathbf{I} \\ \mathbf{N}_l \end{bmatrix}_{(m+n,n)} = \mathbf{Q}\mathbf{R},$$

where  $\mathbf{Q}$  is unitary,  $\mathbf{R}$  is upper-triangular,  $\mathbf{Q} \in \mathbb{R}^{(m+n) \times (m+n)}$ ,  $\mathbf{R} \in \mathbb{R}^{(m+n) \times (n)}$ .



**Fig. 3.** PSNR results of proposed projector and original projector in A+. The red line shows PSNR performance of our method employing with proposed projector. The green one shows the performance of our method employing with original projector. (Color figure online)

Our problem now becomes:

$$\begin{aligned}
 (\mathbf{QR})\mathbf{x} &= \begin{bmatrix} \mathbf{O} \\ \mathbf{f}_l^t \end{bmatrix}_{(m+n,1)}, \\
 \mathbf{R}\mathbf{x} &= \hat{\mathbf{Q}} \begin{bmatrix} \mathbf{O} \\ \mathbf{f}_l^t \end{bmatrix}_{(m+n,1)} \\
 &= [\hat{\mathbf{Q}}_n \ \hat{\mathbf{Q}}_m] \begin{bmatrix} \mathbf{O} \\ \mathbf{f}_l^t \end{bmatrix}_{(m+n,1)} \\
 &\Rightarrow \mathbf{R}\mathbf{x} = \hat{\mathbf{Q}}_m \mathbf{f}_l^t, \\
 \mathbf{y} &= \hat{\mathbf{Q}}_m \mathbf{f}_l^t, \\
 \mathbf{R}\mathbf{x} &= \mathbf{y},
 \end{aligned} \tag{6}$$

where,  $\hat{\mathbf{Q}} = \mathbf{Q}^*$ , and  $\hat{\mathbf{Q}}_m$  is the last  $m$ th columns of  $\hat{\mathbf{Q}}$ ,  $\mathbf{Q}^*$  is conjugate transpose of  $\mathbf{Q}$ , and  $\mathbf{R}\mathbf{x} = \mathbf{y}$  can be solved by substitution method. The performance comparison between normal equation method based and our method based projector is shown in Fig. 3.

### 3 Experiments

We now comprehensively analyze the performance of our proposed NRM in relation to its design parameters and benchmark it in quantitative and qualitative comparison with A+ and other state-of-the-art methods.

We use the training set of images as proposed by Yang *et al.* [16], Timofte *et al.* [15] and by Zeyde *et al.* [17]. However we use a different way to harvest patches from these images. Timofte *et al.* [15] repeatedly harvested dense patches

**Table 1.** Performance of x2, x3, and x4 magnification in terms of averaged PSNR (dB), SSIM and execution time (s) on data set Set5, Set14 and BSD100. Best results in red and runner-up in blue.

data set	s	Bicubic			Zeyde			NE+LLE			A+			SRCNN			RFL			Proposed		
		PSNR	SSIM	Time	PSNR	SSIM	Time	PSNR	SSIM	Time	PSNR	SSIM	Time	PSNR	SSIM	Time	PSNR	SSIM	Time	PSNR	SSIM	Time
Set5	2	33.66	0.9382	0.0	35.78	0.9563	2.5	35.77	0.9560	4.1	36.55	0.9611	0.8	36.34	0.9590	3.0	36.55	0.9585	1.1	<b>36.65</b>	<b>0.9614</b>	1.6
	3	30.39	0.8802	0.0	31.90	0.9075	1.1	31.84	0.9064	1.9	32.59	0.9139	0.5	32.39	0.9141	3.0	32.45	0.9162	1.0	<b>32.67</b>	<b>0.9202</b>	0.8
	4	28.42	0.8246	0.0	29.69	0.8565	0.7	29.61	0.8540	1.1	30.28	0.8737	0.3	30.09	0.8669	3.2	30.13	0.8680	0.8	<b>30.40</b>	<b>0.8700</b>	0.5
Set14	2	30.23	0.9415	0.0	31.81	0.9611	5.0	31.76	0.9620	8.5	32.28	0.9649	1.6	32.18	0.9637	4.9	32.27	0.9442	2.3	<b>32.39</b>	<b>0.9649</b>	3.5
	3	27.54	0.8587	0.0	28.67	0.8859	2.4	28.60	0.8868	3.9	29.13	0.8940	0.9	29.00	0.8910	5.0	29.03	0.8923	1.8	29.20	0.8946	1.7
	4	26.00	0.7838	0.0	26.88	0.8159	1.5	26.81	0.7322	2.4	27.32	0.8281	0.6	27.20	0.8210	5.2	27.21	0.8247	1.3	<b>27.42</b>	<b>0.8300</b>	1.1
B100	2	29.32	0.8338	0.0	30.40	0.8682	3.6	30.41	0.8708	6.1	30.77	0.8773	1.1	<b>31.14</b>	<b>0.8847</b>	3.4	31.13	0.8838	2.5	30.83	0.8772	2.3
	3	27.15	0.7364	0.0	27.87	0.7695	1.8	27.85	0.7713	2.9	28.18	0.7791	0.6	28.21	0.7800	3.4	28.21	0.7805	2.3	<b>28.23</b>	<b>0.7820</b>	1.1
	4	25.92	0.6673	0.0	26.51	0.6968	1.0	26.47	0.6974	1.5	26.77	0.7085	0.4	26.71	0.7022	3.5	26.74	0.7054	2.1	<b>26.83</b>	<b>0.7105</b>	0.7

by means of image pyramid. Because NRM can group a set of dense patches by reconstruction, we employ the Augmented Data set proposed by Timofte *et al.* in [13], which is a more general sparse data set, and harvest it once. To compare with A+ as fairly as possible, we also trained A+ on the Augmented Data set with the same harvest configuration. However, this configuration degraded A+ quality results. So in the following we use the original configurations of A+.

Note that Set5 and Set14 contain respectively 5 and 14 commonly used images for super-resolution evaluation. B100 aka Berkeley Segmentation Dataset is the B100 data set proposed by Timofte *et al.* in [15]. We use the same LR path features as Zeyde *et al.* [17] and Timofte *et al.* [15].

We compare with the following six methods which share the same training data set: standard bicubic upsampling method, the efficient sparse coding method of Zeyde *et al.* [17], neighbor Embedding with Locally Linear Embedding (referred to as NE+LLE) [1], Adjusted Anchored Neighborhood Regression (referred to as A+) of Timofte *et al.* [15], Convolutional Neural Network Method (referred to as SRCNN) of Dong *et al.* [4] and Fast and Accurate Image Upscaling with Super-Resolution Forest (referred to as RFL) of Schuler *et al.* [12].

### 3.1 Results

In order to assess the quality of our proposed method, we tested on 3 datasets (Set5, Set14, B100) used by Timofte *et al.* [15] for 3 upscaling factors (x2, x3, x4) in the same CPU (Intel Core i7 4750HQ 2 GHz) and memory (8 Gb). Considering quality and time cost, we use dictionary with 4096 atoms and a neighborhood size of 2048. The method of Zeyde *et al.*, NE+LLE, the similarity to Chang *et al.* [1], and A+ is set up with its common parameters. SRCNN and RFL are training on the same training data set proposed by Timofte *et al.* leading to a decrease compared to their best performance reported in articles. We report quantitative PSNR and (structural similarity) SSIM results, as well as running times for our bank of methods. In Table 1 we summarize the quantitative results.

In Table 1 we show the averaged PSNR, SSIM and execution times of the benchmark. NRM almost obtains the best PSNR values, around 0.12 dB higher across all scale and data set when compare to the most related algorithm A+. We also outperform some very recent methods (SRCNN and RFL) which are less competitive when trained on the same 91 images training data set. In the terms of computation time, our algorithm is very slightly slower than A+ but still faster than all other methods.

## 4 Conclusion

In this paper we present a new method for regression-based SR that is built on a novel neighbor reconstruction method (NRM). Via manipulations on anchored points and corresponding neighborhoods, NRM can reconstruct new points which are more closer to anchor point on the assumed manifold. Our contributions are: (1) a new sample reconstruction method with application to regression-based SR; (2) Supported by matrix QR decomposition, we design a more condition-number-stable regressor to compute effective result under closer neighborhood situation. Our results confirm the effectiveness of this approach using various accepted benchmarks, where we clearly outperform the current state-of-the-art. Finally, when the harvested samples are sparse on the manifold, NRM can still construct much closer points and perform well.

**Acknowledgments.** This work is supported by National Natural Science Foundation of China (Grant No. 61402389) and the Fundamental Research Funds for the Central Universities (No. 20720160073).

## References

1. Chang, H., Yeung, D.-Y., Xiong, Y.: Super-resolution through neighbor embedding. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, p. I. IEEE (2004)
2. Cui, Z., Chang, H., Shan, S., Zhong, B., Chen, X.: Deep network cascade for image super-resolution. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 49–64. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-10602-1\\_4](https://doi.org/10.1007/978-3-319-10602-1_4)
3. Dai, D., Timofte, R., Van Gool, L.: Jointly optimized regressors for image super-resolution, vol. 34, pp. 95–104. Wiley Online Library (2015)
4. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **38**(2), 295–307 (2016)
5. Dong, W., Zhang, L., Shi, G., Xiaolin, W.: Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization. *IEEE Trans. Image Process.* **20**(7), 1838–1857 (2011)
6. Fattal, R.: Image upsampling via imposed edge statistics. *ACM Trans. Graph. (TOG)* **26**(3) (2007)
7. Freeman, W., Jones, T., Pasztor, E.: Example-based super resolution. *IEEE Trans. Comput. Graph. Appl.* **22**(2), 56–65 (2002)
8. Kim, K.I., Kwon, Y.: Single-image super-resolution using sparse regression and natural image prior. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(6), 1127–1133 (2010)
9. Prez-Pellitero, E., Salvador, J., Torres, I.: PSyCo: manifold span reduction for super resolution. In: CVPR (2016)
10. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. *Science* **290**(5500), 2323–2326 (2000)
11. Salvador, J., Ruiz-Hidalgo, J., Rosenhahn, B., et al.: Half hypersphere confinement for piecewise linear regression. In: IEEE Winter Conference on Applications of Computer Vision, WACV, pp. 1–9 (2016)

12. Schulter, S., Leistner, C., Bischof, H.: Fast and accurate image upscaling with super-resolution forests. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3791–3799 (2015)
13. Timofte, R., Rothe, R., Van Gool, L.: Seven ways to improve example-based single image super resolution. In: CVPR (2016)
14. Timofte, R., De Smet, V., Van Gool, L.: Anchored neighborhood regression for fast example-based super-resolution. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1920–1927 (2013)
15. Timofte, R., De Smet, V., Van Gool, L.: A+: adjusted anchored neighborhood regression for fast super-resolution. In: Cremers, D., Reid, I., Saito, H., Yang, M.-H. (eds.) ACCV 2014. LNCS, vol. 9006, pp. 111–126. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-16817-3\\_8](https://doi.org/10.1007/978-3-319-16817-3_8)
16. Yang, J., Wright, J., Huang, T.S., Ma, Y.: Image super-resolution via sparse representation. *IEEE Trans. Image Process.* **19**(11), 2861–2873 (2010)
17. Zeyde, R., Elad, M., Protter, M.: On single image scale-up using sparse representations. In: Boissonnat, J.-D., et al. (eds.) *Curves and Surfaces 2010*. LNCS, vol. 6920, pp. 711–730. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-27413-8\\_47](https://doi.org/10.1007/978-3-642-27413-8_47)
18. Zhang, K., Tao, D., Gao, X., Li, X., Xiong, Z.: Learning multiple linear mappings for efficient single image super-resolution. *IEEE Trans. Image Process.* **24**(3), 846–861 (2015)





# An Efficient Method for Boundary Detection from Hyperspectral Imagery

Suhad Lateef Al-Khafaji<sup>(✉)</sup>, Jun Zhou<sup>(✉)</sup>, and Alan Wee-Chung Liew

School of Information and Communication Technology,  
Griffith University, Nathan, Australia

suhad.al-khafaji@griffithuni.edu.au, jun.zhou@griffith.edu.au

**Abstract.** In this paper, we propose a novel method for efficient boundary detection in close-range hyperspectral images (HSI). We adopt different spectral similarity measurements to construct a sparse spectral-spatial affinity matrix that characterizes the similarity between the spectral responses of neighboring pixels within a local neighborhood. After that, we adopt a spectral clustering method in which the eigenproblem is solved and the eigenvectors of smallest eigenvalues are calculated. Morphological erosion is then applied on each eigenvector to detect the boundary. We fuse the results of all eigenvectors to obtain the final boundary map. Our method is evaluated on a real-world HSI dataset and compared with three alternative methods. The results exhibit that our method outperforms the alternatives, and can cope with several scenarios that methods based on color images can not handle.

**Keywords:** Boundary detection · Edge detection  
Spectral clustering · Spectral feature extraction

## 1 Introduction

In computer vision, boundary in an image can be defined as sudden change of brightness, color or texture between two neighboring regions. Boundary detection is an important process in image processing, with many research introduced for both gray-level and color images. Typically, boundary detection method can be divided into two categories: edge detection and segmentation. Traditional edge detection methods include for instance Canny edge detector [1] and gradient methods [2] which are most successful in discriminating neighboring regions with high contrast. Image segmentation methods determine boundaries between regions by partitioning an image into separate classes [3]. Recently, researchers have adopted various complex cues for estimating boundaries in images rather than just using color or brightness [4]. Some methods attempt to combine different cues to extract global or low level features to learn the boundary in color images [5].

Compared with color images, hyperspectral images (HSI) are more informative by providing spectral responses at each pixel [6]. An HSI can be considered

as an image cube where the third dimension indexes many band images of contiguous spectral wavelengths. As a result, a pixel in a hyperspectral image is a vector whose dimension equals to the number of spectral bands. The image contains valuable spectral information that can be used to account for pixel variability, similarity and discrimination [7]. On the other hand, processing of HSI is a challenging task. Due to the imaging mechanism, hyperspectral images are sensitive to noises and normally have lower spatial resolution compared to color images. Furthermore, the multi-band nature makes the amount of data to be processed very large in HSI [8].

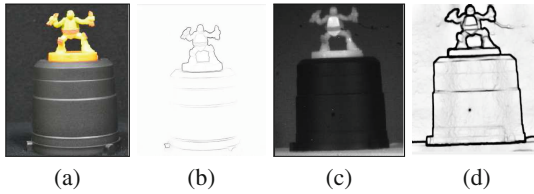
There are very few attempts on edge detection or boundary detection in HSI. Most existing work were proposed for hyperspectral remote sensing [9,10], and can not be readily applied to computer vision scenarios. In computer vision, change of illumination, shape of objects, image resolution, and layout of objects have to be considered in image analysis [11]. These factors are normally ignored in remote sensing where the objects are far from the imaging sensor. For close-range HSI, Al-Khafaji et al. [12] proposed a statistical spectral information based method for boundary detection. This method calculates the probability of occurrence of boundary pixels based on spectral-spatial features. The probability is estimated using kernel density estimator (KDE). Although this work is effective for close-range HSI boundary detection, calculating statistical information and the KDE step lead to high computational cost.

The main objective of this work is to exploit spectral information to detect boundary effectively, especially for cases where methods based on color images can not handle. At the same time, our goal also includes developing an efficient method to handle large amount of data, and addressing the drawback in [12]. Our method is based on the observation that local neighboring pixels within the same object have similar spectral responses, but pixel pairs on the boundary have different spectral responses even though they may have similar color and texture. Thanks to the power of spectral information, utilizing spectral responses is adequate to distinguish boundary pixels, without the need of using complex features as in [12]. Furthermore, instead of calculating the probability of pixels occurrence which is of high computational cost, we use a simple and fast spectral similarity measure between the spectral responses of neighbouring pixels to identify pixels on the boundary. The spectral similarity is used to construct a weighted spectral-spatial adjacency matrix. Then eigenproblem for the matrix is solved by calculating the eigenvectors that correspond to the smallest eigenvalues. After that, we perform morphological erosion on each eigenvector image and fuse the results for all eigenvector images to form the boundary map.

In Fig. 1(b), we show the result of boundary detection on Fig. 1(a) which is an RGB image. The method from Isola et al. [13] missed the boundaries of the black base since its color is similar to the background. On the contrary, these boundaries are preserved using our method as shown in Fig. 1(d) when a hyperspectral image in Fig. 1(c) is used.

In summary, the novel contribution of this paper comes from three aspects.

- This is one of the first works on boundary detection from HSI, especially in a close-range imaging setting.
- We adopt the spectral response (spectral signature) to recognize pixels on object boundary, since the spectral contrast of neighboring pixels straddle on boundary is very clear. Thus we do not need to extract high level features.
- We use efficient spectral similarity measures to calculate the affinity matrix, so as to avoid the high computational cost of KDE as in [12].
- Instead of Gaussian derivative filter which was used in [12], we adopt morphological erosion to produce the boundary map from the calculated eigenvectors images.



**Fig. 1.** Boundary detection results for HSI and RGB images: (a) An RGB image of target objects; (b) Boundary detection result on (a) using method in [13]. (c) An HSI of target objects; (d) Boundary detection result on (c) using our method. We can observe the missing boundaries of the base in the RGB image while they are preserved in the HSI result. (Color figure online)

The rest of this paper is organized as follows. Section 2 presents our proposed method for boundary detection from HSI. In Sect. 3, we introduce the newly collected dataset and present the experimental results and comparison with other methods. Finally, conclusions are drawn in Sect. 4.

## 2 Boundary Detection Method

Our boundary detection method has two main stages, sparse spectral-spatial affinity graph construction to generate eigenvectors, and boundary map construction by applying morphological erosion on the generated eigenvectors. In the graph construction step, we adopt the similarity between the spectral responses of neighbouring pixels to construct a weighted spectral-spatial adjacency matrix  $W_{ss}$ . The spectral response vector at each pixel  $pv_i$  is considered as a vertex, where  $i = 1, 2, \dots, n$  and  $n$  is the number of pixels in HSI. The edges between them are weighted using the spectral similarity measurement. Then eigenproblem for matrix  $W_{ss}$  can be solved by calculating the eigenvectors that correspond to the smallest eigenvalues. After that, we perform morphological erosion on each eigenvector image to extract the boundaries, and then fuse the results for all eigenvector images to form the boundary map.

## 2.1 Spectral-Spatial Affinity Graph

Spectral clustering is the core of this stage. It is based on a similarity graph  $G = (V, E)$ , where the relationship between data points in  $V$  is characterized by edges in  $E$ . In an HSI, the similarity between two neighboring pixels can be identified based on the similarity of their spectral responses. Objects made of different materials normally have distinctive spectral responses, even though their color or texture may be similar. In addition, regions with different colors and textures within a single object will provide different spectral responses. Figure 2 depicts that pixels belong to the same region have similar spectral responses while pixel pairs straddle boundary have different spectral responses. Therefore, boundaries in HSI can be defined by any sudden changes in the spectral response where these include any changes in material, color and texture, so we can extract spectral, spatial, or spectral-spatial boundaries in the HSI.

Therefore, the first step in this work is to construct a sparse spectral-spatial similarity matrix utilizing spectral features. For an HSI  $H \in \mathbb{R}^{N \times M \times B}$  where  $N$  and  $M$  are the spatial dimensions and  $B$  is the spectral dimension (number of bands), all image pixels  $i$  and  $j$  within spatial distance of radius  $r$  ( $r = 5$  in our experiments) are represented as spectral vectors, and are used to form the vertices of a connected graph. An edge in the graph correspond to the affinity between two vertices  $pv_i$  and  $pv_j$ :

$$W_{ss_{ij}} = \exp(-F(pv_i, pv_j)/c) \quad (1)$$

where  $F(pv_i, pv_j)$  is the spectral similarity between  $pv_i$  and  $pv_j$  and  $c$  is a parameter to control the magnitude of similarity. Different similarity measurements can be used to model the relationship between two spectral vectors [14]. In this work, we compare four different spectral measurements: spectral angle mapper (SAM), spectral gradient angle (SGA), normalized spectral Euclidean distance (NED), and spectral information divergence (SID) [15, 16]. SAM is defined as follows:

$$SAM(pv_i, pv_j) = \arccos\left(\frac{\sum_{k=1}^B pv_{ik}pv_{jk}}{\sqrt{\sum_{k=1}^B pv_{ik}^2} \sqrt{\sum_{k=1}^B pv_{jk}^2}}\right) \quad (2)$$

SGA is built on top of SAM, and can be calculated as:

$$SGA(pv_i, pv_j) = SAM(SG_{pv_i}, SG_{pv_j}) \quad (3)$$

where  $SG_{pv_i}$  is the spectral gradient for  $pv_i$  and is defined as:

$$SG_{pv_i} = [pv_{i_2} - pv_{i_1}, pv_{i_3} - pv_{i_2}, \dots, pv_{i_B} - pv_{i_{B-1}}] \quad (4)$$

The calculation of NED is straightforward:

$$NED(pv_i, pv_j) = e(N_{pv_i}, N_{pv_j}) \quad (5)$$

where  $e$  is the Euclidean distance and  $N_{pv_i} = pv_i/\sqrt{pv_i}$  is the normalized pixel vector. Finally, SID is defined as:

$$SID(pv_i, pv_j) = D(pv_i \| pv_j) + D(pv_j \| pv_i) \quad (6)$$

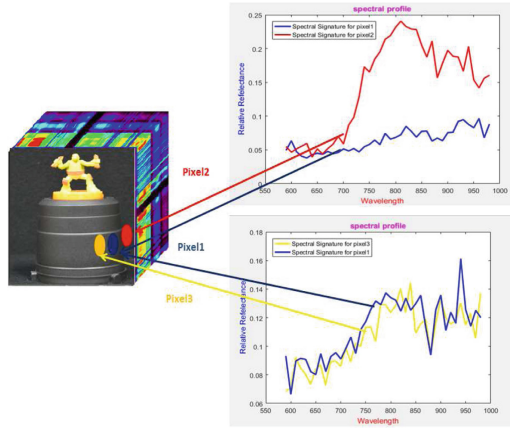
where

$$D(pv_i || pv_j) = \sum_{k=1}^B p_k \log(p_k/q_k) \tag{7}$$

and

$$D(pv_j || pv_i) = \sum_{k=1}^B q_k \log(q_k/p_k) \tag{8}$$

are derived from two probability vectors  $p = (p_1, p_2, \dots, p_B)^T$  and  $q = (q_1, q_2, \dots, q_B)^T$  for the spectral responses of vectors  $pv_i$  and  $pv_j$ , where  $p_k = pv_{ik} / \sum_{l=1}^B pv_{il}$  and  $q_k = pv_{jk} / \sum_{l=1}^B pv_{jl}$ .



**Fig. 2.** Spectral responses for three neighboring pixels. The blue and yellow pixels belong to the same region (black base) and have similar spectral responses. The blue and red pixels are on different sides of the boundary and have different spectral responses, although their black color are similar. (Color figure online)

When the similarity measurements are ready, the sparse spectral-spatial affinity matrix is constructed. Then we have the following eigenproblem:

$$(D - Wss)v = \lambda Dv \tag{9}$$

where  $D_{ii} = \sum_{j \neq i} Wss_{ij}$ , and  $\lambda$  is the eigenvalue corresponding to eigenvector  $v$ . We compute the generalized eigenvectors that are corresponding to the smallest  $m$  eigenvalues of system in Eq. (9). Due to the large size of the affinity matrix  $Wss$ , computing the eigenvectors can be very costly even though  $Wss$  is sparse. We therefore used a method in [17] for fast eigenvector computation.

It has been observed that each eigenvector can be treated as an image and contains different boundary information. Figure 3 presents examples of four eigenvectors. In practice, we use  $m = 50$ . The first row shows the images of eigenvectors that are calculated using our method while the second row shows the

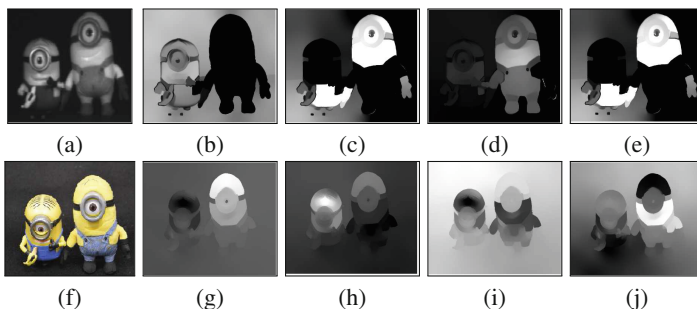
eigenvector images for the RGB image using method in [13]. We can observe that using spectral-spatial affinity matrix is more effective without missing parts since each eigenvector image contains boundary information. Whereas, using affinity matrix based on texture and color features produces eigenvector images with some missing parts. For instance, the hand of minion is always missing in all eigenvector images as shown in Fig. 4(g)–(j), this is because the hands and the background screen have very similar black colour and texture, though they are made of different materials.

## 2.2 Morphological Erosion

In the traditional spectral clustering [18], each pixel is associated with a descriptor of length  $m$  created from elements of  $m$  eigenvectors. Then clustering algorithms such as K-means can be applied to divide the image to clusters. Arbelaez et al. [3] pointed out that according to the smooth variation of eigenvectors, using K-means algorithm can break up the large uniform image regions and this will produce incorrect segmentation. Therefore, they convolved each eigenvector image with Gaussian derivative filters at 8 different orientations to overcome the smooth variations.

However, the smooth variation issue of the eigenvectors still affects the results since some parts of the boundaries are too smooth and makes the boundary unclear. Furthermore, convolving each eigenvector with 8 orientations can be of high computational cost. Thus, we adopt a simple morphological erosion method to extract image boundaries from the eigenvector images.

Mathematical morphology is a simple non-linear technique in image processing, which deals directly with geometric shape of objects [19]. It is considered as an efficient tool for shape information extraction and it has two basic operations: erosion and dilation. In our method, we use the basic erosion operation with a  $3 \times 3$  flat square structural element (kernel) to go through the grayscale eigen



**Fig. 3.** (a) An HSI image. (b)–(e) The first four generalized eigenvectors resulting from spectral clustering using spectral-spatial affinity matrix. (f) An RGB image of the same scene as (a). (g)–(j) The first four generalized eigenvectors resulting from spectral clustering using affinity matrix based on color and texture features [13]. (Color figure online)

images. The erosion operation calculates the minimum of the pixels in each pixel neighborhood defined by the structuring element. Thus, its function is similar to many other image filters such as the median filter and the Gaussian filter. At this point, erosion can be used to remove pixels on object boundaries. After that, subtracting the eroded image from the original image will produce the image boundary:

$$Ev_i = v_i \ominus s \quad (10)$$

where  $s$  is the structural element. Finally, we subtract the result of each  $Ev_i$  from the original eigenvector  $v_i$  and then sum the subtraction results to form the boundary map:

$$Bss = \sum_{i=1}^m (v_i - Ev_i) \quad (11)$$

### 3 Experimental Results

Our experiments were conducted on an HSI dataset collected by the Spectral Imaging Lab at Griffith University. This dataset was collected using a hyperspectral camera which consists of a Brimrose acousto-optical tunable filter (AOTF) and a highly sensitive visible to infrared camera. The HSI dataset consists of 30 images of indoor and outdoor scenes with various objects such as toys, boxes, plants and buildings. In addition, we captured RGB images of the same views using an RGB camera positioned next to the hyperspectral camera. Figure 5 shows sample images in this dataset.

Each HSI has 61 spectral bands with wavelengths ranging from 400 nm to 1000 nm at 10 nm spectral resolution. The quality of the HSIs is affected by many factors such as incident lighting condition, camera focusing, and distance between the camera and the objects. Moreover, the signal to noise ratio is low in some bands although our camera is highly sensitive. Therefore, we removed some very noisy bands from the images. Consequently, 40 spectral bands from 590 nm to 980 nm with 10 nm spectral resolution were used. However, the remaining 40 spectral bands still suffer from artifacts which affect the quality of image. Thus, a 3D Gaussian smoothing filter was used to reduce the noises in both spectral and spatial domains of the HSI.

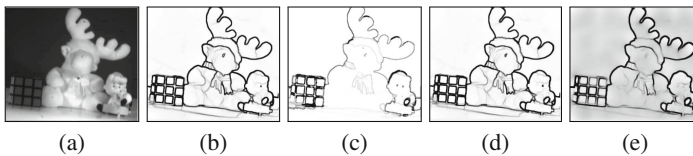
To demonstrate the effectiveness of our method, we also adopted several hyperspectral images of natural scenes collected by Foster et al. [20]. These HSIs were captured using a low-noise Peltier-cooled digital camera that provides a spatial resolution of  $1344 \times 1024$  and 33 spectral bands with wavelengths ranging from 400 nm to 720 nm with bandwidth of 10 nm at 550 nm, decreasing to 7 nm at 400 nm and increasing to 16 nm at 720 nm. The last row of Fig. 5 shows an outdoor sample image in this dataset.

We compared our method with three boundary detections approaches. The first two methods are based on RGB images [13, 21]. The method from Isola et al. [13] adopted the statistical information of pixel features (color and texture) to detect image boundary. While method from Leordeanu et al. [21] combined

low-level static cues (pixel intensity) with depth and occlusion cues to generalize image boundary. Another comparison was conducted against the method in [12] which was proposed for HSI boundary detection. In implementing our model, we set  $c = 0.01$  for Eq. (1), and  $m = 50$  to get the smallest  $m$  eigenvalues in Eq. (9). For all other methods, we set relevant parameters according to the original papers. Furthermore, all RGB images and HSIs were scaled into same spatial size ( $400 \times 400$  pixels).

### 3.1 Performance of Different Similarity Measurements

In this experiment, we compared the performance of different spectral similarity measurements that were used to calculate the affinity matrix. Figure 4 exhibits the results of using SAM, SGA, NED and SID, respectively. From Fig. 4(b) and (d), we can observe that SAM and NED give similar results, which are better than the outcome of SGA and SID in Fig. 4(c) and (e) respectively. Result of SAM and NED demonstrate high correlation indicating the presence of material. SAM describes the similarity from the perspective of vector direction (angle). Since pixels on boundary have significant variation in direction, large angle between two spectral responses on boundary indicates low similarity. Whereas, NED takes into account the difference of brightness between two pixel vectors [15]. Therefore, the values of these measures indicate the level of changes between neighboring pixels on boundary.



**Fig. 4.** Results on different spectral similarity measures for constructing the affinity matrices. (a) The original HSI; (b) Result of using SAM; (c) Result of using SGA; (d) Result of using NED; (e) Result of using SID.

### 3.2 Method Comparison

In this experiment, we compared the performance of the proposed method and three alternative approaches in the literature. The results are reported in Fig. 5. We did a qualitative evaluation since the ground truth is not available. Our method with spectral measurement SAM produces better results than all other methods. From the first and third rows of Fig. 5, we can see that there are some missing boundaries in the results from the RGB images such as minion's hat in the fourth and fifth columns in the first row of Fig. 5. This is due to the fact that the boundary and the background have similar color and texture, making the detection methods fail to distinguish them. On the contrary, thanks to the exploitation of spectral information in the HSI, the boundaries are well



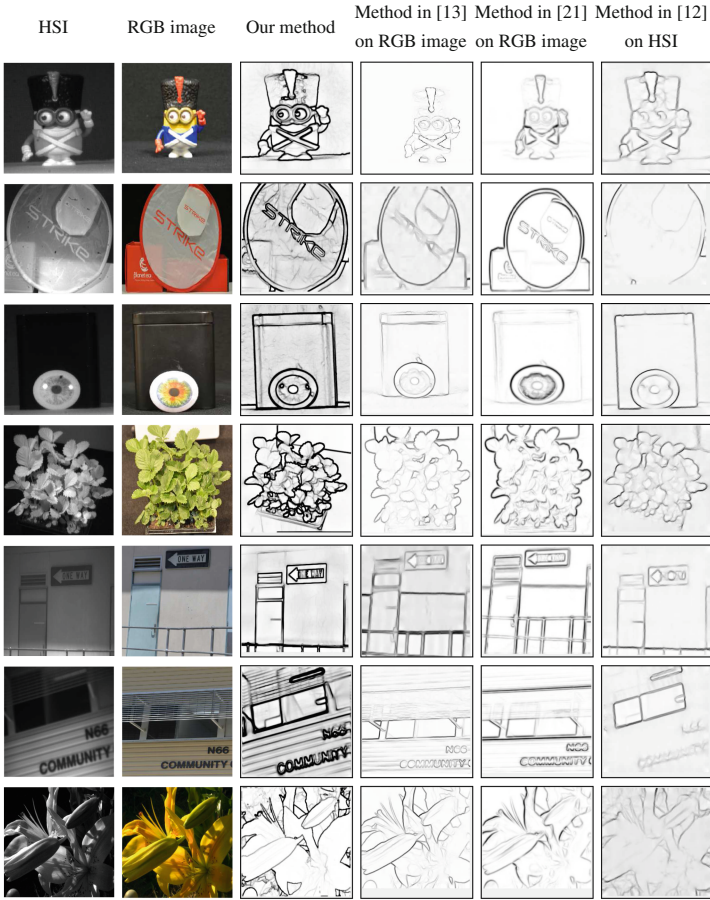


Fig. 5. Boundary detection results on sample images. The first four rows are indoor HSIs, while the last three rows are outdoor HSIs. (Color figure online)

preserved using our method. Another example can be shown in the sixth row of Fig. 5. We can observe that there is a light behind the window which is partially occluded by the metal screen. The boundary of the light is clearly displayed in the results from hyperspectral images, but not RGB images. This again validates the effectiveness of our method and using spectral data.

Comparing with the results in the fifth row of Fig. 5 for HSI boundary detection, our method achieves the best performance. It uses the spectral response directly to detect boundary pixels, fully exploring the spectral information. Furthermore, using morphological erosion on the obtained eigenvectors instead of using Gaussian derivative filter produces much thicker boundaries, and thus improves the final results. In addition, our method has much lower computational cost than [12]. Our method takes by average around 2 min to process an

image while the method in [12] takes around 15 min per image, when running the program using Matlab on a laptop with an Intel Core i5 processor and 8 GB memory.

## 4 Conclusion

In this paper, we present a novel method for boundary detection from HSIs based on exploiting the spectral features. Spectral responses can be used to discriminate two neighboring pixels of different materials with distinct reflectance. Our method effectively combines the output of the similarity matrix with the morphological erosion. It produces robust results on a collected HSI dataset. Comparing with existing boundary detection approach on HSIs, the proposed method has demonstrated high efficiency with better detection quality.

**Acknowledgement.** The work of Suhad Lateef Al-khafaji was partially supported by Iraqi Ministry of Higher education and scientific research, Al-Nahrain University, Iraq.

## References

1. Canny, J.: A computational approach to edge detection. In: Readings in Computer Vision, pp. 184–203. Elsevier (1987)
2. Haralick, R.M.: Digital step edges from zero crossing of second directional derivatives. In: Readings in Computer Vision, pp. 216–226. Elsevier (1987)
3. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(5), 898–916 (2011)
4. Hallman, S., Fowlkes, C.: Oriented edge forests for boundary detection. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1732–1740 (2015)
5. Yang, K., Gao, S., Guo, C., Li, C., Li, Y.: Boundary detection using double-opponency and spatial sparseness constraint. *IEEE Trans. Image Process.* **24**(8), 2565–2578 (2015)
6. Liang, J., Zhou, J., Qian, Y., Wen, L., Bai, X., Gao, Y.: On the sampling strategy for evaluation of spectral-spatial methods in hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **55**(2), 862–880 (2016)
7. Tong, L., Zhou, J., Qian, Y., Bai, X., Gao, Y.: Nonnegative matrix factorization based hyperspectral unmixing with partially known endmembers. *IEEE Trans. Geosci. Remote Sens.* **54**(11), 6531–6544 (2016)
8. Bai, X., Guo, Z., Wang, Y., Zhang, Z., Zhou, J.: Semi-supervised hyperspectral band selection via spectral-spatial hypergraph model. *IEEE J. Sel. Top. Appl. Earth Observations Remote Sens.* **8**(6), 2774–2783 (2015)
9. van der Werff, H., van Ruitenbeek, F., van der Meijde, M., van der Meer, F., de Jong, S., Kalubandara, S.: Rotation-variant template matching for supervised hyperspectral boundary detection. *IEEE Geosci. Remote Sens. Lett.* **4**(1), 70–74 (2007)
10. Chen, C., Guo, B., Wu, X., Shen, H.: An edge detection method for hyperspectral image classification based on mean shift. In: *International Congress on Image and Signal Processing*, pp. 553–557 (2014)

11. Gu, L., Robles-Kelly, A., Zhou, J.: Efficient estimation of reflectance parameters from imaging spectroscopy. *IEEE Trans. Image Process.* **22**(9), 3648–3663 (2013)
12. Al-Khafaji, S.L., Zia, A., Zhou, J., Liew, A.W.: Material based boundary detection in hyperspectral images. In: *The International Conference on Digital Image Computing: Techniques and Applications*, pp. 1–7 (2017)
13. Isola, P., Zoran, D., Krishnan, D., Adelson, E.H.: Crisp boundary detection using pointwise mutual information. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) *ECCV 2014*. LNCS, vol. 8691, pp. 799–814. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-10578-9\\_52](https://doi.org/10.1007/978-3-319-10578-9_52)
14. Wang, K., Yong, B., Gu, X., Xiao, P., Zhang, X.: Spectral similarity measure using frequency spectrum for hyperspectral image classification. *IEEE Geosci. Remote Sens. Lett.* **12**(1), 130–134 (2015)
15. Robila, S., Gershman, A.: Spectral matching accuracy in processing hyperspectral data. In: *International Symposium on Signals, Circuits and Systems*, vol. 1, pp. 163–166 (2005)
16. van der Meer, F.: The effectiveness of spectral similarity measures for the analysis of hyperspectral imagery. *Int. J. Appl. Earth Obs. Geoinf.* **8**(1), 3–17 (2006)
17. Arbeláez, P., Pont-Tuset, J., Barron, J.T., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 328–335 (2014)
18. Shi, J., Malik, J.: Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(8), 888–905 (2000)
19. Amer, A.: New binary morphological operations for effective low-cost boundary detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **17**(2), 1–13 (2002)
20. Foster, D., Amano, K., Nascimento, S., Foster, M.: Frequency of metamerism in natural scenes. *J. Opt. Soc. Am. A* **23**, 2359–2372 (2006)
21. Leordeanu, M., Sukthankar, R., Sminchisescu, C.: Efficient closed-form solution to generalized boundary detection. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) *ECCV 2012*. LNCS, vol. 7575, pp. 516–529. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-33765-9\\_37](https://doi.org/10.1007/978-3-642-33765-9_37)

# **Graph-Theoretic Methods**



# Bags of Graphs for Human Action Recognition

Xavier Cortés<sup>(✉)</sup>, Donatello Conte, and Hubert Cardot

LiFAT, Université de Tours, Tours, France  
{xavier.cortes,donatello.conte,  
hubert.cardot}@univ-tours.fr

**Abstract.** Bags of visual words are a well known approach for images classification that also has been used in human action recognition. This model proposes to represent images or videos in a structure referred to as bag of visual words before classifying. The process of representing a video in a bag of visual words is known as the encoding process and is based on mapping the interest points detected in the scene into the new structure by means of a codebook. In this paper we propose to improve the representativeness of this model including the structural relations between the interest points using graph sequences. The proposed model achieves very competitive results for human action recognition and could also be applied to solve graph sequences classification problems.

## 1 Introduction

Human action recognition in video sequences has become a necessary task in several applications such as human-robot interaction, autonomous driving, surveillance systems and many others. However, an accurate recognition performance of human actions is a very challenging task.

Bags of Visual Words (BoVW) used before for images classification [1–3] have been shown as a successful way to address the problem of human action recognition [4–7]. The key idea of this approach is to map the interest points detected in a human action video in a representative structure referred to as BoVW taking into account its features.

In order to improve the representativeness of the BoVW model, we propose to include in the representation the structural relations between the interest points instead of evaluating the points individually.

A typical way to represent structured objects is by means of graphs. Graphs are defined by a set of nodes (interest points in our case) and edges (connections between the nodes) and they have become very important in pattern recognition. Graphs have been successfully applied in several domains such as cheminformatics, bioinformatics and computer vision among others [8–10]. We propose to represent human actions by means of graph sequences.

It is important to remark that most of the fields in which graphs have been applied in pattern recognition, are based on single graph representations estimating graph distances [11] or classifying graphs [12]. However, dynamic or time dependent problems are very common in several pattern recognition applications. For instance signal processing, study of chemical interactions, proteins folding, evaluation of

diseases behaviors on populations or the human action recognition problem addressed in this paper can be represented by streams of graphs evolving through the temporal dimension. Due to this, another important contribution of this paper is to present a method to classify graph sequences.

The paper is organized as it follows, in Sect. 2, we introduce the necessary definitions to understand the paper, in Sect. 3, we present a model to transform a video in a graphs sequence, in Sect. 4, we present a classification model for graph sequences, finally, in Sects. 5 and 6, we show the experimental results and the conclusions.

## 2 Definitions

In this section we introduce some definitions necessary to contextualize and understand the paper.

### 2.1 Attributed Graph

Formally, we define an attributed graph as a quadruplet  $g = (\Sigma_v, \Sigma_e, \gamma_v, \gamma_e)$ , where  $\Sigma_v = \{v_i | i = 1, \dots, n\}$  is the set of attributed nodes,  $\Sigma_e = \{e_{ij} | i, j \in 1, \dots, n\}$  is the set of edges connecting pairs of nodes,  $\gamma_v$  is a function that maps the nodes to their attributed values and  $\gamma_e$  maps the edges.

### 2.2 Graph Edit Distance

The Graph Edit Distance (GED) [13, 14] defines a distance model between two attributed graphs  $g^p$  and  $g^q$  through the minimum amount of distortion required to transform  $g^p$  into  $g^q$ . To do this, a set of edit operations of insertion, deletion, and substitution of nodes and edges are required. Edit cost functions are typically used to evaluate the level of distortion of each edit operation. A sequence of edit operations that completely transform  $g^p$  into  $g^q$  is referred to as *editpath* between  $g^p$  and  $g^q$ . The total cost of the edit operations included in an *editpath* could be considered as a distance between  $g^p$  and  $g^q$ . Note, that there are several *editpaths* between two graphs depending on the edit operations we use to do the transformation. Formally, GED is defined as the minimum cost under all possible *editpaths*  $T$ .

$$\text{GED}(g^p, g^q) = \min_{\gamma \in T} \text{EditCost}(g^p, g^q, \gamma) \quad (1)$$

### 2.3 Sub-optimal Graph Edit Distance Computation

Optimal algorithms for computing the GED are based on complex search procedures. These procedures typically explore a wide range of possible *editpaths* between  $g^p$  and  $g^q$  selecting the smaller in terms of total cost. The main drawback of these methods is that they are very complex in terms of computational cost.

In order to reduce its computational complexity, the problem of minimizing the GED has been sub-optimally reformulated in [15, 16]. However in these works the

problem still has a considerable computational complexity. More recently, in [17], the authors propose a quadratic time approximation of GED based on the Hausdorff matching algorithm. For a better understanding of the details of this algorithm we encourage to read the original paper [17].

## 2.4 Graph Sequences

We define a graphs sequence  $G = \{g_1, \dots, g_n, \dots, g_N\}$  as a stream of graphs representing the evolution through  $N$  different states, represented by graphs, of a single object.

## 2.5 Bags of Graphs

Bags of Words (BoW) are a kind pattern representation model that has been used for several years in language processing [18] and more recently as BoVW in image [1–3] and video classification [4–7]. A BoW is a global object descriptor consisting of different bins counting the mappings between the components of the represented object and the words of a codebook. We can distinguish three fundamental parts in this model, the first one is the codebook generation, the second one is the encoding procedure to embed the objects in a BoW and the last one is the classification algorithm.

In [19], the authors introduced Bags-of-Graphs (BoG), a particular type of BoW to encode digital objects into histograms based on local structures defined by graphs. The authors propose to use the BoG to encode single graph representations as proteins, letters or images.

Inspired by [19], in this paper, we propose to use a BoG to encode and classify graph sequences.

# 3 Representing Human Actions by Means of Graph Sequences

We propose to represent each video by means of a graphs sequence. The original video is divided into splits of a predefined number of consecutive frames and each split is represented by a graph. The process consists of the following steps. First, we extract the interest points that appear in the frames of the original video. To do this, we propose to use a Spatio-Temporal Interest Point detector (STIP) [20] that can be seen as an extension of the Harris detector [21] but taking into account the temporal dimension. Next, we divide the original video into splits of consecutive frames and we group the interest points within the split where they have been detected. We build one graph per split. To do this we find the Convex Hull [22] on the spatial coordinates where the interest points have been detected to find which points are the vertexes of the smallest polygon enveloping all the points detected in the same split. Applying this method, we filter the interest points using only the vertexes and consequently we limit the cardinality and the density of the graph representations reducing also the computational complexity of the problem. Moreover, we assume that for human action recognition tasks, the peripheral interest points are more informative than the internal interest

points. To feature the nodes we propose to use the Histogram of Optical Flows (HOF) [23] of the corresponding interest points as attributes. Finally, to represent the structure, we use the sides of the Convex Hull polygon. If two nodes belong to the ends of the same side, we connect them by an edge. Figure 1 shows the process described in this section.

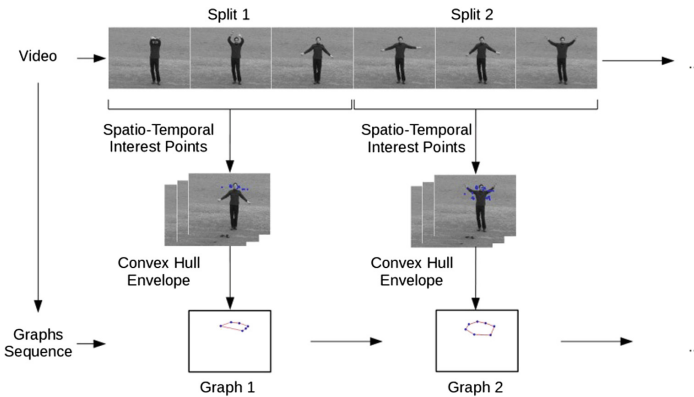


Fig. 1. Representing human action videos by means of graph sequences.

#### 4 Graph Sequences Classification Using Bags of Graphs

We propose to use BoGs representations (introduced in Sect. 2.5) to encode graph sequences into histograms, mapping the graphs of the sequence to the graphs represented in a codebook.

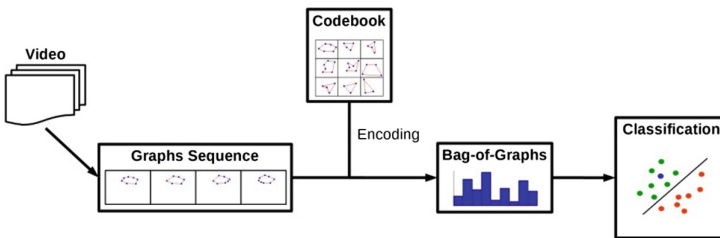


Fig. 2. Human action classification based on BoG scheme.

Figure 2 shows the general scheme of this classification model. First, we find the corresponding graphs sequence from a human action video, this procedure is described in Sect. 3, next, we encode the graphs sequence in a BoG using a graph codebook and finally we perform the classification. In Sect. 4.1 we propose a method to build a graph codebook of representative graphs from a training set while in Sect. 4.2 we explain how to encode a graphs sequence in a BoG given a graph codebook. Finally, in Sect. 4.3 we detail how to classify BoGs.



#### 4.1 Generation of Graph Codebooks by Means of Graph Clustering

Graph codebooks are graph collections used to encode graph sequences in BoGs. A representative selection of graphs in the codebook is crucial for the performance of the model. To build a graph codebook we propose to follow a multi-level clustering approach based on the k-means algorithm [24] similar to the one presented in [7]. This approach proposes to build the codebook by means of clustering the interest points extracted from a set of training videos. The clustering is performed at different levels in order to reduce the computational complexity of the process and to be more robust to the noise. In our model we propose to cluster graphs instead of interest points. In the first level we cluster the graphs of the sequences extracted from the training videos (Sect. 3) in order to select a subset of representative graphs per sequence while in the second level we cluster the output graphs of the first level to select the action representatives. The codebook is finally built attaching the output graphs of the second level in a single structure. In Fig. 3, we show a general scheme of the codebook generation process.

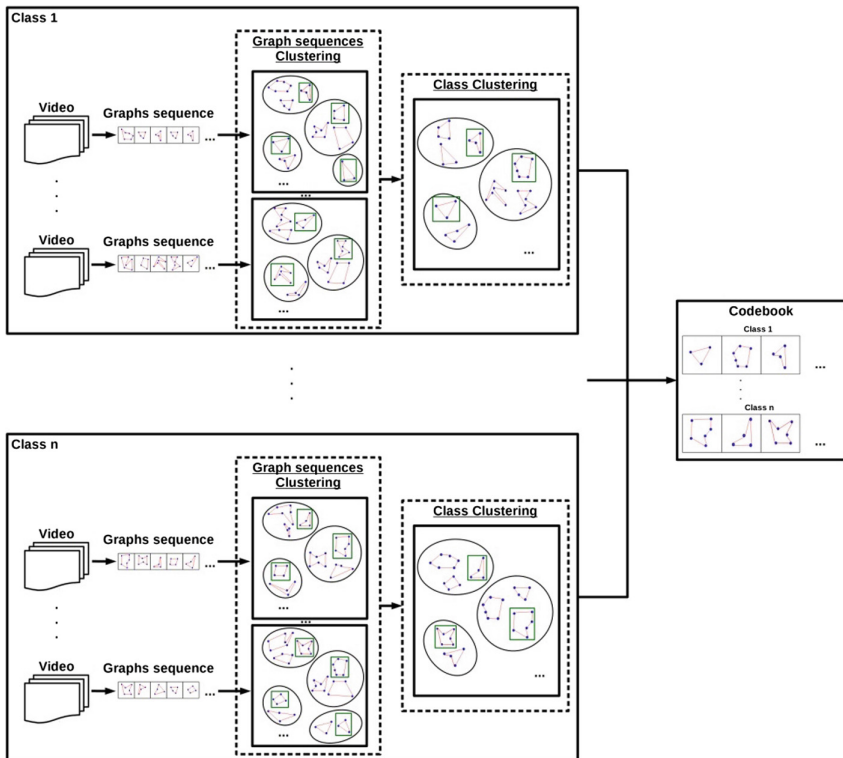


Fig. 3. Graph codebook generation scheme.

The graph clustering problem has been addressed by several authors in the literature as in [25, 26] because is not trivial given the computational complexity of the GED. We followed a similar approach to the one presented in [26] to perform the graph clustering. The authors propose to embed the graphs before applying the k-means clustering algorithm. The embedding problem aims to convert graphs into another structure to make more manageable the operations. There are different methods to solve the graph embedding problem as in [26, 27]. In our model, we propose to embed graphs in  $n$ -dimensional vector spaces. The values of the embedded vector are filled by taking the GED between the graphs we are embedding to each one of the graphs in the set we are clustering. Once all the graphs have been embedded the k-means algorithm is applied on the embedded representations. The outputs of the k-means algorithm are  $k$ -centroids in the vector space corresponding to  $k$ -clusters. Finally, as clusters representatives, we select the graphs whose embedded representations are the closest to the centroids found by the k-means algorithm.

## 4.2 Bags of Graphs Encoding

The encoding is the procedure to represent a graphs sequence in a BoG. The BoG is a histogram divided in different bins. Each bin corresponds to one of the graphs in the codebook. We propose to follow a soft-approach [28] updating each bin according the GED between the graph of the sequence that we are mapping and the corresponding graph in the codebook. Formally:

A BoG  $\in \mathbb{R}^J$  is defined as a vector of  $J$  bins representing a graphs sequence where  $N$  is the number of graphs in a sequence  $G = \{g_1, \dots, g_n, \dots, g_N\}$  and  $J$  is the number of representative graphs in a codebook  $W = [w_1, \dots, w_j, \dots, w_J]$ .

We encode the graphs sequence  $G$  into each bin  $\text{BoG}_j$  of the BoG using the graph codebook  $W$  as follows:

$$\text{BoG}_j = \sum_{n=1}^N u(g_n, w_j) \quad (2)$$

Where:

$$u(g_n, w_j) = \frac{e^{(-\beta \cdot \text{GED}(g_n, w_j))}}{\sum_{k=1}^J e^{(-\beta \cdot \text{GED}(g_n, w_k))}} \quad (3)$$

Where  $\beta$  a parameter to control the softness of the assignment and GED is the distance function between two graphs.

## 4.3 Bags of Graphs Classification

To perform the classification we propose to train one linear SVM [29] per class targeting the BoGs to the corresponding classes of the training videos. The trained SVMs are used to identify if the BoGs representing the videos that we want to classify belong to a class or not.

## 5 Experiments

The aim of our experiments is to empirically evaluate the performance of the model classifying videos of humans performing different actions. We tested the experiments on the KTH [30] dataset, which is commonly used in the human action recognition domain to compare results.

The dataset consist of 599 videos corresponding to 6 different action classes. The actions are performed by 25 actors in 4 different scenarios. The testing set consists of the videos performed by the first 9 actors and the training set by the videos performed by the next 16 actors.

We build the codebook using the graph sequences generated from the training videos following a multilevel clustering approach as we describe in Sect. 4.1. In the first level, we select a sample of the 10% of graphs that appear in the original sequence and in the second level we select 50 graphs for each human action. Finally, given 6 actions and 50 graphs per action we build a graph codebook of 300 graphs. To build the graphs sequence as we described in Sect. 3 we divide the original video into splits of 50 frames. The parameter  $\beta$  of the encoder (Sect. 4.2) is fixed to 0.75. Due to its good balance in terms of computational complexity and classification accuracy, we have used the Hausdorff-GED (Sect. 2.3) as the GED measure and the Clique centrality [31] as the costs function penalizing the structural dissimilarities.

**Table 1.** Accuracy results of our method and other state-of-the-art models following a similar experimental configuration.

Method	Accuracy
Elshourbagy et al. [7]	97.7
Bilinski et al. [32]	96.3
Bregonzio et al. [33]	94.3
Wang et al. [6]	92.1
Klaser et al. [34]	91.4
Laptev et al. [5]	91.8
Zhang et al. [35]	91.3
Dollár et al. [36]	81.2
Our method	96.5

In Table 1 we show a comparison between our method and other recently presented results following a similar experimental configuration. The values correspond to the average classification accuracy percentage achieved on each human action using a linear SVM classifier per class. Our method is the second best with respect to the state-of-the-art presented in the table, so proving the competitiveness of our solution.

Figure 4 shows some sample graphs appearing in the original sequence and the corresponding BoG belonging to different action classes. We observe how BoG representing videos of the same action class tend to be more similar.

Class	Sample Graphs of the Sequence					Bag of Graphs
<i>Boxing 1</i>						
<i>Boxing 2</i>						
<i>Handwaving 1</i>						
<i>Handwaving 2</i>						
<i>Handclapping 1</i>						
<i>Handclapping 2</i>						
<i>Walking 1</i>						
<i>Walking 2</i>						
<i>Jogging 1</i>						
<i>Jogging 2</i>						
<i>Running 1</i>						
<i>Running 2</i>						

Fig. 4. Sample graphs and BoG of different human actions in the KTH dataset.

## 6 Conclusions

The main purpose of the paper is to present a method for human action recognition based on BoG. To perform this task, we propose a model consisting of two main parts. The first part consists of transforming of the human action video in a sequence of graphs. The second part is to encode the sequence of graphs in a BoG before classifying. We experimentally prove that our method is competitive compared with some of the best state-of-the-art results. Another relevant contribution of our paper is the idea to use the BoG model to classify graph sequences. For future works we consider to evaluate the performance of our model using different GED measures and to address new problems represented by graph sequences using our classification model.

**Acknowledgments.** This work is part of the LUMINEUX project supported by a Region Centre-Val de Loire (France). We gratefully acknowledge Region Centre-Val de Loire for its support.

## References

1. Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: Workshop on Statistical Learning in Computer Vision, ECCV, Prague, vol. 1, no. 1–22, pp. 1–2 (2004)
2. Yang, J., Yu, K., Gong, Y., Huang, T.: Linear spatial pyramid matching using sparse coding for image classification. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 1794–1801. IEEE (2009)
3. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality constrained linear coding for image classification. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3360–3367. IEEE (2010)
4. Niebles, J.C., Wang, H., Fei-Fei, L.: Unsupervised learning of human action categories using spatial-temporal words. *Int. J. Comput. Vis.* **79**(3), 299–318 (2008)
5. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2008, pp. 1–8. IEEE (2008)
6. Wang, X., Wang, L., Qiao, Y.: A comparative study of encoding, pooling and normalization methods for action recognition. In: Lee, K.M., Matsushita, Y., Rehg, J.M., Hu, Z. (eds.) ACCV 2012. LNCS, vol. 7726, pp. 572–585. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-37431-9\\_44](https://doi.org/10.1007/978-3-642-37431-9_44)
7. Elshourbagy, M., Hemayed, E., Fayek, M.: Enhanced bag of words using multilevel k-means for human activity recognition. *Egypt. Inform. J.* **17**(2), 227–237 (2016)
8. Mahé, P., Vert, J.-P.: Graph kernels based on tree patterns for molecules. *Mach. Learn.* **75**(1), 3–35 (2009)
9. Qi, X., Wu, Q., Zhang, Y., Fuller, E., Zhang, C.-Q.: A novel model for DNA sequence similarity analysis based on graph theory. *Evol. Bioinform.* **7**, 149–158 (2011)
10. Conte, D., Foggia, P., Sansone, C., Vento, M.: Thirty years of graph matching in pattern recognition. *Int. J. Pattern Recogn. Artif. Intell.* **18**(3), 265–298 (2004)
11. Li, T., Dong, H., Shi, Y., Dehmer, M.: A comparative analysis of new graph distance measures and graph edit distance. *Inf. Sci.* **403–404**, 15–21 (2017)
12. Solé-Ribalta, A., Cortés, X., Serratos, F.: A Comparison between structural and embedding methods for graph classification. *SSPR/SPR* **2012**, 234–242 (2012)
13. Sanfeliu, A., Fu, K.: A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man Cybern.* **13**, 353–362 (1983)
14. Bunke, H., Allermann, G.: Inexact graph matching for structural pattern recognition. *Pattern Recogn. Lett.* **1**(4), 245–253 (1983)
15. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**(4), 950–959 (2009)
16. Serratos, F.: Speeding up fast bipartite graph matching through a new cost matrix. *Int. J. Pattern Recogn. Artif. Intell.* **29**(2), 1550010 (2015)
17. Fischer, A., Suen, C.Y., Frinken, V., Riesen, K., Bunke, H.: Approximation of graph edit distance based on Hausdorff matching. *Pattern Recogn.* **48**(2), 331–343 (2015)
18. Harris, Z.S.: Distributional structure. *Word* **10**(2–3), 146–162 (1954)

19. Silva, F.B., Werneck, R.d.O., Goldenstein, S., Tabbone, S., Torres, R.d.S.: Graph-based bag-of-words for classification. *Pattern Recogn.* **74**, 266–285 (2018)
20. Laptev, I.: On space-time interest points. *Int. J. Comput. Vis.* **64**(2–3), 107–123 (2005)
21. Harris, C., Stephens, M.: A combined corner and edge detector. In: *Alvey Vision Conference*, Manchester, UK, vol. 15, no. 50, pp. 147–151 (1988)
22. Andrew, A.M.: Another efficient algorithm for convex hulls in two dimensions. *Inf. Process. Lett.* **9**(5), 216–219 (1979)
23. Pers, J., Sulic, V., Kristan, M., Perse, M., Polanec, K., Kovacic, S.: Histograms of optical flow for efficient representation of body motion. *Pattern Recogn. Lett.* **31**(11), 1369–1376 (2010)
24. Hartigan, J.A., Wong, M.A.: Algorithm AS 136: a k-means clustering algorithm. *J. Roy. Stat. Soc. Ser. C (Appl. Stat.)* **28**(1), 100–108 (1979)
25. Galluccio, L., Michel, O.J.J., Comon, P., Hero III, A.O.: Graph based k-means clustering. *Sig. Process.* **92**(9), 1970–1984 (2012)
26. Ferrer, M., Valveny, E., Serratos, F., Bardají, I., Bunke, H.: Graph-based k-means clustering: a comparison of the set median versus the generalized median graph. In: Jiang, X., Petkov, N. (eds.) *CAIP 2009*. LNCS, vol. 5702, pp. 342–350. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-03767-2\\_42](https://doi.org/10.1007/978-3-642-03767-2_42)
27. Bunke, H., Riesen, K.: Improving vector space embedding of graphs through feature selection algorithms. *Pattern Recogn.* **44**(9), 1928–1940 (2011)
28. Liu, L., Wang, L., Liu, X.: In defense of soft-assignment coding. In: *2011 IEEE International Conference on IEEE Computer Vision (ICCV)*, pp. 2486–2493 (2011)
29. Campbell, C., Ying, Y.: Learning with support vector machines. *Synth. Lect. Artif. Intell. Mach. Learn.* **5**(1), 1–95 (2011)
30. Schuldt, C., Laptev, I., Caputo, B.: Recognizing human actions: a local SVM approach. In: *Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004*, vol. 3, pp. 32–36. IEEE (2004)
31. Serratos, F., Cortés, X.: Graph edit distance: moving from global to local structure to solve the graph-matching problem. *Pattern Recogn. Lett.* **65**, 204–210 (2015)
32. Bilinski, P., Bremond, F.: Statistics of pairwise co-occurring local spatio-temporal features for human action recognition. In: Fusiello, A., Murino, V., Cucchiara, R. (eds.) *ECCV 2012*. LNCS, vol. 7583, pp. 311–320. Springer, Heidelberg (2012). [https://doi.org/10.1007/978-3-642-33863-2\\_31](https://doi.org/10.1007/978-3-642-33863-2_31)
33. Bregonzio, M., Xiang, T., Gong, S.: Fusing appearance and distribution information of interest points for action recognition. *Pattern Recogn.* **45**(3), 1220–1234 (2012)
34. Klaser, A., Marszałek, M., Schmid, C.: A spatio-temporal descriptor based on 3D-gradients. In: *Bmvc 2008-19th British Machine Vision Conference*, pp. 275:1–10. British Machine Vision Association (2008)
35. Zhang, Z., Hu, Y., Chan, S., Chia, L.-T.: Motion context: a new representation for human action recognition. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) *ECCV 2008*. LNCS, vol. 5305, pp. 817–829. Springer, Heidelberg (2008). [https://doi.org/10.1007/978-3-540-88693-8\\_60](https://doi.org/10.1007/978-3-540-88693-8_60)
36. Dollár, P., Rabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features. In: *2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 65–72. IEEE (2005)



# Categorization of RNA Molecules Using Graph Methods

Richard C. Wilson<sup>(✉)</sup> and Enes Algul

University of York, York, UK  
{richard.wilson,enes.algul}@york.ac.uk

**Abstract.** RNA molecules are a group of biologically active molecules which have a similar structure to DNA. Graph-based methods for classification have shown promise on other biological compounds such as protein. In this paper, we investigate the use of graph representations of RNA, graph-feature based methods and their role in classifying RNA into particular categories. We describe a number of possible graph representations of RNA structure and how useful information can be encoded in the graph. We show how graph-kernel and graph-feature methods can be used to provide descriptors for the molecules. Finally, on a moderately-sized database of 419 RNA structures, we explore how these methods can be used to classify RNA into high-level categories provided by the biological context or function of the molecules. We find that graph descriptors give state-of-the-art performance on sequence classification, but that the graph elements of the description do not add useful information above the base-sequence.

## 1 Introduction

Graphs have proved to be a valuable representation in bioinformatics and chemoinformatics. They have been used to represent networks of protein interactions and chemical structures for example. Structural pattern recognition and machine learning can then be used to categorize and classify new data from examples. This approach has been particularly successful in the classification of molecular databases where biological activity can be inferred from the data [1].

In contrast, in other biologically relevant structures such as DNA, the graph-based approach is not so crucial and it is the sequence encoded in the DNA bases which is important in pattern recognition problems. Here string-matching is typically used. Proteins, which are constructed from the base-sequence of DNA, are particularly interesting because they exhibit string-like properties from the base-sequence, relational properties from the local contact of parts, and geometry from the overall shape.

RNA molecules are very similar to DNA in the sense that they are constructed from a sequence of nucleotides and can encode information. However, they only consist of one strand, not two as in DNA, and hence can fold into complex patterns like proteins. RNA therefore also manifests string, relational and geometric properties like proteins.

In this paper we will explore the use of pattern recognition methods for the problem of categorizing RNA into high-level biologically-relevant classes. We will then use these tools to explore whether sequence, geometry and relational structure actually indicates the function of a particular RNA.

## 2 Related Work

RNA is a molecule which has been relatively little-studied using graph methods. Most methods rely on sequence or 2D [2] and 3D [3] molecule shape. Recent methods of trying to understand RNA have focussed on a structural classification of the molecule into parts, based on a two-dimensional representation of the molecule. This is similar to the structural representation of a protein where the primary structure is the amino-acid sequence and the secondary structure is a classification of 3D shape, such as  $\alpha$ -helix or  $\beta$ -sheet. For example, STRAND [2] classifies the structural features of RNA using a secondary structural analyzer [4]. This results in a detailed secondary structure classification into motifs such as stem, pseudoknot, hairpin loops, bulge loops and so on.

The similarity of DNA structures is generally determined by a sequence similarity, since the sequence is a code for the functionality of the DNA. The sequence similarity is determined by sequence alignment, for example by the Needleman-Wunsch algorithm [5]. This allows for nucleotide substitution and gaps in the sequence in a similar fashion to the string edit distance. The similarity is dependent on the number of sites which are the same. The same method can be applied to RNA, but RNA molecules have direct biological function, so it is not clear whether the sequence is more important than the shape.

Of course, RNA is a chemical structure and is amenable to methods used to classify chemical compounds [6]. These methods typically involve deriving a set of features or constructing a kernel based on structural elements such as paths and walks. The state-of-the-art methods for chemical structure classification are based on approximate edit distance and graph kernels. Riesen and Bunke [7] use a cost matrix derived from the local edit distance followed by bipartite matching. Borgwardt et al. [8] represent proteins using a graph derived from secondary structure and vertex-based chemical properties. They utilise the random walk kernel to measure similarities between graphs. Mahé and Vert [9] propose a tree-counting kernel for the recognition of chemical structure graphs. These methods are generally not strongly dependent on sequence or geometry as the first is not relevant for general chemical structures and the second is difficult to encode in a graph structure. Other methods for classifying proteins have mainly focused on graph matching, where there is an explicit correspondence between parts. These methods can include sequence and geometry as the specific arrangement of parts is known, for example [10].

## 3 Preliminaries

A graph  $G = (V, E)$  is an object consisting of a set  $V$  of vertices and a set  $E \subseteq V \times V$  of edges. The vertices represent sub-parts. A pair of vertices  $(u, v)$



is in the edge set  $E$  is there is some pairwise relationship between them. The vertex  $u$  is said to be adjacent to  $v$  ( $u \sim v$ ) if  $(u, v) \in E$ . We are concerned here only with undirected graphs where the edges are bidirectional. Each vertex  $u$  has a label associated with it via a labelling function  $l(u) \in L$  where  $L$  is some set of labels.

A *path* on a graph is a sequence of vertices  $(u_1, u_2, \dots, u_n)$  such that each consecutive pair are joined by an edge  $(u_i, u_{i+1}) \in E$  and no vertex is repeated in the sequence, except for possibly the first and last. The length of a path is the number of edges traversed,  $n - 1$ . A *simple cycle* is a path where the first and last vertices are the same.

A *graph feature* is a number representing some property of the graph and dependent only on the structure of the graph (not on vertex or edge order). For example, the number of edges or the maximum degree are both graph features. A *graph kernel*  $K(G_i, G_j)$  is a similarity function between a pair of graphs which satisfies the kernel properties of symmetry and positive-definiteness. Both paths and cycles can be used to construct graph features and kernels, as we explain later.

## 4 Data

The RNA set was compiled and labelled by Klosterman et al. [3]. It consists of 419 structures of RNA extracted from the Protein Data Bank (PDB) and the Nucleic Acid Database (NDB). Each RNA molecule is classified in a hierarchical scheme from more general labels to the more specific. The labelling is summarised in Table 1. The top-level categories include transfer RNAs, ribosomal RNAs, ribozymes and small nuclear RNAs (snRNAs) among others, and also some synthetic RNAs without biological function. Subcategories are also provided. Transfer RNAs are grouped into initiator tRNA, elongator tRNAs, synthetase complexes, etc. For each of these RNAs, structural and geometric information is available, in particular the nucleotide sequence and the atom positions within the molecule. The matched base-pairs are inferred from the sequence and proximity of the bases.

## 5 Representation of RNA

RNA consists of a sequence of nucleotides which are usually coded from DNA. The nucleotide contains a fixed backbone forming the polymer and a variable base from the set adenine (A), cytosine (C), guanine (G) and uracil (U). The sequence of bases are joined in a chain, forming the primary structure of the molecule. As in DNA, the bases bond with each other, in the pairs A-U and C-G. RNA consists of a single chain, but because of the base pairing, the chain can fold to bond with itself, creating a 3D shape. Figure 1 illustrates the main features of an RNA molecule. The 3D plot at the top of this figure illustrated the complex 3D shape produced by folding and pair-bonding. In most RNA studies, the shape is represented by some secondary structure (bottom) which is

**Table 1.** Label hierarchy. The numbers in brackets represent the number of examples in the dataset.

Level 1	Level 2	Level 3
molNaturalOccur (387)	Ribosomal_RNA(132) Ribozyme(45) telomeraseRNA(2) Genetic_Control_Element(8) Viral_RNA(100) Transfer_RNA(67) SnRNA(9) SRP_RNA(10) MessengerRNAoriginal(12) tmRNA(2)	49 classes...
Evolved_(SELEX)_RNA(31) <Undefined>(1)	Aptamer(31) <undefined>(1)	

essentially a 2D schematic diagram of the molecule. The main features of most RNA are stems, formed by a sequence of pair-bonds, and loops of unbonded bases. Other structures, such as pseudoknots, are possible but not considered here.

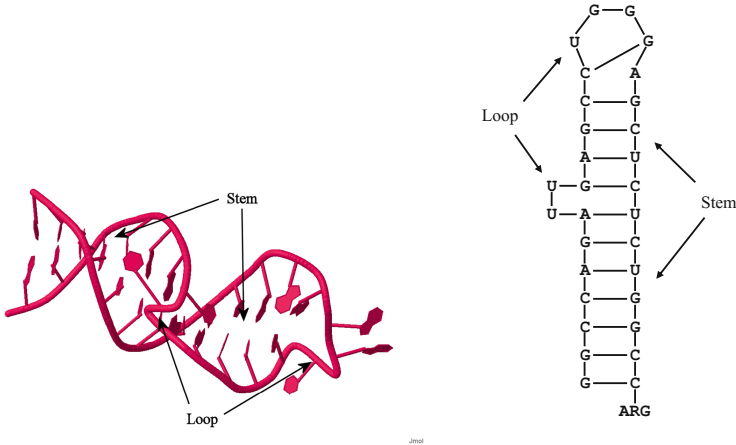
Our goal is to represent this molecule as a graph which encodes all three aspects of RNA (sequence, structure and geometry). The primary structure is easily encoded as a set of vertices, each vertex representing a base. The vertices are optionally labelled by the base type (A, C, G, U). We will explore the effect of this labelling later. The chain is represented by connecting adjacent bases in the sequence into a string.

The pairing of the bases is a little more complex. We consider two bases to be paired if they are compatible (A-U or C-G) and their end-points are within  $4\text{\AA}$  of each other. Matched pairs are denoted by an edge in the graph between the corresponding vertices.

Loops can be identified by their geometry. Referring to Fig. 1(top), the bases in a loop point outwards because they are un-bonded and repulsion forces them further apart. We detect this using the difference between the backbone spacing and the inter-base spacing of the nucleotides. If the latter is larger, the site is part of a loop.

## 6 Comparison of RNA

In this work, we use graph comparison methods to categorise the RNA data into a set of high-level categorisations. Because of the size of the database, we focus on feature and kernel-based methods, as opposed to matching methods which are computationally expensive. While these methods are essentially relational,



**Fig. 1.** 3D and secondary structure of the HIV-2 TAR RNA ‘1akx’

the encoding of structure and geometry into the representation allows us to also consider these factors.

## 6.1 Kernel-Based Methods

A graph kernel  $K(G_i, G_j)$  is essentially a similarity function for graphs which satisfies the kernel properties of symmetry and positive-definiteness. In contrast to feature-based methods which are applied to single graphs, the kernel computes a similarity by comparing each pair of graphs. It therefore sensitive to more specific differences between the pairs of graphs, but at the expense of more computation. Kernels are popular in machine learning and commonly used with kernel machines such as the support vector machine.

The graph kernel values for each pair of graphs can be formed into a kernel matrix  $K_{ij} = K(G_i, G_j)$ . Once the kernel matrix has been computed, kernel embedding can be used to project the graphs into a feature space representing the kernel. This allows the use of any standard machine learning method directly on the features. Of course this process requires the full dataset to be available.

**Weisfeiler-Lehmen Optimal Assignment Kernel.** The Weisfeiler-Lehmen Optimal Assignment Kernel (WL-OA) is a recently proposed graph kernel [11] which has shown great promise in machine learning problems for graphs. The WL-OA kernel is computed using the Weisfeiler-Lehmen label refinement process [12]. At each step of the refinement process, the labels from neighboring vertices are gathered to construct a new label for the central vertex. As the iteration proceeds, information from a larger part of the graph is integrated into the vertex label. We then employ the method described in [11] to compute the similarity of the optimal match between the vertices of two graphs.

## 6.2 Feature-Based Methods

We consider the application of two feature-based methods for graph comparison to the problem of comparing the RNA structures. While both methods were developed in the context of constructing graph kernels, they are based on counting similar paths, and so have an explicit embedding into the labelled path space. We use this embedding here as a feature space for the graphs to improve computational efficiency.

**Shortest Path Embedding.** The shortest path kernel (SPK) [13] evaluates the shortest paths between each pair of vertices in a graph. The shortest paths are labelled in some way (for example by length), and then the similarity between two graphs is evaluated as the number of such paths which are the same between the two graphs. In our RNA application, we label each path by the path length and the two vertex labels at the start- and end-points of the path. This kernel has an explicit embedding into feature space as a histogram over the labelled paths. The RNA molecules are therefore represented as counts of the number of shortest paths which have a particular length and start/end labels.

**All Paths and Cycles Embedding.** The all paths and cycles kernel (APC) [14] is a recently proposed kernel which counts all possible paths and simple cycles in the graph, rather than just the shortest path of the SPK. Again this kernel admits a direct embedding in the feature space of labelled paths. In this case, the paths are labelled by the method described in [14], which is a histogram over the numbers of each label type which appear in the path.

## 6.3 Sequence-Based Methods

As a comparison, we also employ a standard sequence-based method typically employed for DNA comparison. The RNA is encoded by the nucleotide sequence, essentially a string of A, C, G, U (with occasional non-standard bases). The strings are aligned using the Needleman-Wunsch algorithm [5] and the p-distance between them,  $0 \leq p \leq 1$ , is the fraction of sites which differ. The distance between two RNA sequences is given by the Jukes-Cantor score

$$d = -\frac{3}{4} \log\left(1 - \frac{4}{3}p\right). \quad (1)$$

This results in a distance matrix  $\mathbf{D}$  containing the distance between all pairs. We use multi-dimensional scaling to embed these distances in feature space. Since the alignment-distance is not metric, the Gram matrix contains negative eigenvalues and we cannot obtain an exact embedding. Rather than discard the negative eigenvalues, we use the absolute value [15], i.e.

$$\begin{aligned} \mathbf{K} &= -\frac{1}{2} (\mathbf{I} - \mathbf{J}/n) \mathbf{D} (\mathbf{I} - \mathbf{J}/n) \\ \mathbf{K} &= \mathbf{U} \Lambda \mathbf{U}^T \\ \mathbf{X} &= \mathbf{U} \sqrt{|\Lambda|} \end{aligned} \quad (2)$$

## 6.4 Classification

We classify the molecules into one of 12 classes (following the Level-2 classification in Table 1) using the following procedure. Firstly, the described methods are used to generate a feature set for each molecule. In the case of the kernel method WL-OA, kernel embedding is used to obtain the implicit feature space. Then we apply PCA to remove redundant components with very small variance, for efficiency. We then use random subspace kNN for classification, which we found gives the best results on all our methods.

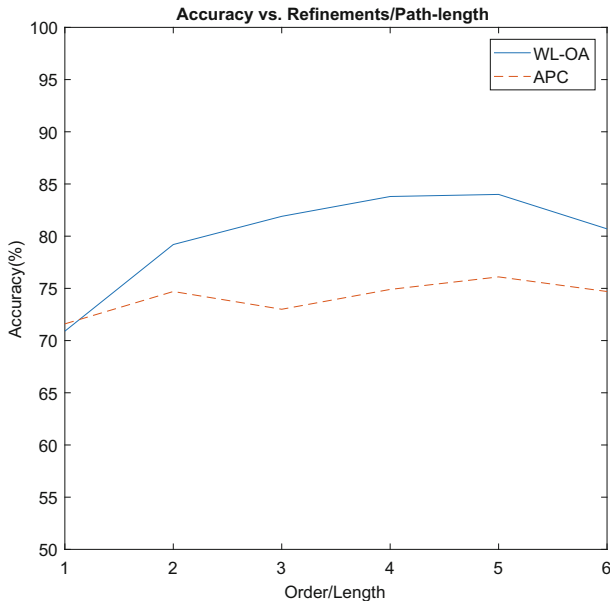
## 7 Results

There are two goals of our analysis. Firstly, we want to establish whether graph-based representations are a suitable method for classifying RNA structure. To this end, we use our database of RNA molecules to classify the structures into the twelve level-2 classifications listed in Table 1. Our second goal is to establish what structural information is important for the classification. We extract three sources of information from the structure; the topology of the graph, the geometry of the shape and the base-sequence. We aim to find out which of these is the most important. Secondly, we wish to discover which of the graph-based methods are most effective on this dataset (Table 2).

**Table 2.** Classification accuracies for the RNA dataset using a variety of methods and representations.

	Sequence only	Topology only	Topology + sequence	Topology + geometry	All
WL-OA	<b>84.0</b>	68.7	81.9	62.5	80.1
SP	77.1	72.3	76.8	67.3	<b>78.3</b>
APC	<b>76.1</b>	56.3	75.7	65.9	-
SA	73.3	-	-	-	-

The methods evaluated are the Weisfeiler-Lehmen Optimal Assignment Kernel (WL-OA), the embedding derived from the shortest path kernel (SP), the all paths and cycles embedding (APC) and the sequence alignment (SA). The SA method operators only on the sequence of nucleotides, as described in the previous section. For each of the other methods, different information is included in the graph. The base graph is simply a path connecting the vertices in sequence order. ‘Sequence’ adds labels to the vertices indicating the nucleotide at each site, and is the graph equivalent of the plain nucleotide sequence. ‘Topology’ adds the cross-link edges indicating matched base-pairs in the structure. ‘Geometry’ includes additional labels on the vertices indicating the local type of secondary structure (stem/loop). In the method SA, the data is purely the string



**Fig. 2.** Effect of changing order for the WL-OA method or maximum path length for the APC method on the sequence-only data.

of nucleotide letters. APC can only accommodate a small number of labels, so we do not run this method with the full label set.

The results show a number of surprising features. Firstly, on the sequence alone, the graph based methods WL-OA and SP outperform sequence alignment even though only sequence information is used. They both produce a richer description of the RNA for the purposes of classification than SA. SA assumes that the sequences are the same, with insertions and deletions, and this does not seem to be the best model for determining the RNA class.

Secondly, the nucleotide sequence is, by far, the best source of information for classifying the RNA in this study. The addition of topological information produces a marginal reduction in classification accuracy, and only SP shows any improvement with the inclusion of all additional information. From a biological standpoint, it seems clear that the shape must have something to do with the function of a strand of RNA, so we can conclude that our simple geometric labelling is insufficient to extract useful information. In Fig. 2 we illustrate the effect of changing the number of refinements in the labelling process for WL-OA when used on the sequence-only graph. Similarly we plot the performance of APC versus the maximum path length used. From the plot it is clear that performance peaks at  $L = 5$  indicating that the sequence information most relevant to the current site is within five bases.

## 8 Conclusion

In this paper, we have described methods for encoding RNA in a graph structure and shown how recent graph comparison methods can be used to measure the similarity of two RNA molecules. We applied machine learning to classify RNA into high level classes. Our best result was a accuracy of 84.0% using the WL-OA kernel on the sequence information only. This compares to a baseline (random guess) accuracy of 20.0%.

Our results demonstrate that graph-based feature and kernel methods improve on sequence alignment for RNA classification. However, the graph elements of the description do not provide additional information for classification problem. Adding links representing matched base-pairs does not improve the accuracy, nor does adding simple descriptors of loops and stems. We believe that more sophisticated descriptions of the structure is needed.

## References

1. Helma, C., Kramer, S.: A survey of the predictive toxicology challenge 2000–2001. *Bioinformatics* **19**, 1179–1182 (2003)
2. Andronescu, M., Bereg, V., Hoos, H.H., Condon, A.: Rna strand: The rna secondary structure and statistical analysis database. *BMC Bioinform.* **9**(1), 340 (2008)
3. Klosterman, P., Tamura, M., Holbrook, S., Brenner, S.: Scorer: a structural classification of RNA database. *Nucleic Acids Res.* **30**, 392–394 (2002)
4. <http://www.rnasoft.ca/strand/>
5. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* **43**(3), 443–453 (1970)
6. Wale, N., Watson, I.A., Karypis, G.: Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowl. Inf. Syst.* **14**, 347–375 (2008)
7. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**(7), 950–959 (2009). 7th IAPR-TC15 Workshop on Graph-based Representations (Gbr 2007). <http://www.sciencedirect.com/science/article/pii/S026288560800084X>
8. Borgwardt, K.M., Ong, C.S., Schoenauer, S., Vishwanathan, S.V.N., Smola, A.J., Kriegel, H.P.: Protein function prediction via graph kernels. *Bioinformatics* **21**, i47–i56 (2005)
9. Mahé, P., Vert, J.-P.: Graph kernels based on tree patterns for molecules. *Mach. Learn.* **75**(1), 3–35 (2009). <https://doi.org/10.1007/s10994-008-5086-2>
10. Rocha, J., Segura, J., Wilson, R.C., Dasgupta, S.: Flexible structural protein alignment by a sequence of local transformations. *Bioinformatics* **25**(13), 1625–1631 (2009). <https://doi.org/10.1093/bioinformatics/btp296>
11. Kriege, N.M., Giscard, P.-L., Wilson, R.C.: On valid optimal assignment kernels and applications to graph classification. In: *Advances in Neural Information Processing Systems*, pp. 1615–1623 (2016)
12. Shervashidze, N., Schweitzer, P., van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-Lehman graph kernels. *J. Mach. Learn. Res.* **12**, 2539–2561 (2011). <http://dl.acm.org/citation.cfm?id=2078187>

13. Borgwardt, K.M., Kriegel, H.: Shortest-path kernels on graphs. In: Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005), Houston, Texas, USA, 27–30 November 2005, pp. 74–81 (2005). <https://doi.org/10.1109/ICDM.2005.132>
14. Giscard, P.-L., Wilson, R.C.: The all-paths and cycles graph kernel. arXiv preprint [arXiv:1708.01410](https://arxiv.org/abs/1708.01410) (2017)
15. Pełkalska, E., Harol, A., Duin, R.P.W., Spillmann, B., Bunke, H.: Non-euclidean or non-metric measures can be informative. In: Yeung, D.-Y., Kwok, J.T., Fred, A., Roli, F., de Ridder, D. (eds.) SSPR /SPR 2006. LNCS, vol. 4109, pp. 871–880. Springer, Heidelberg (2006). [https://doi.org/10.1007/11815921\\_96](https://doi.org/10.1007/11815921_96)





# Quantum Edge Entropy for Alzheimer's Disease Analysis

Jianjia Wang<sup>(✉)</sup>, Richard C. Wilson, and Edwin R. Hancock

Department of Computer Science, University of York, York YO10 5DD, UK  
{jw1157, Richard.Wilson, Edwin.Hancock}@york.ac.uk

**Abstract.** In this paper, we explore how to decompose the global statistical mechanical entropy of a network into components associated with its edges. Commencing from a statistical mechanical picture in which the normalised Laplacian matrix plays the role of Hamiltonian operator, thermodynamic entropy can be calculated from partition function associated with different energy level occupation distributions arising from Bose-Einstein statistics and Fermi-Dirac statistics. Using the spectral decomposition of the Laplacian, we show how to project the edge-entropy components so that the detailed distribution of entropy across the edges of a network can be achieved. We apply the resulting method to fMRI activation networks to evaluate the qualitative and quantitative characterisations. The entropic measurement turns out to be an effective tool to identify the differences in structure of Alzheimer's disease by selecting the most salient anatomical brain regions.

**Keywords:** Alzheimer's disease · Bose-Einstein statistics  
Fermi-Dirac statistics · Network entropy

## 1 Introduction

Functional magnetic resonance imaging (fMRI) has provided a sophisticated means of studying the neuro-pathophysiology associated with Alzheimer's disease (AD) [11]. It maps the network representation to neuronal activity between the various brain regions. The resulting network structure has proved useful in understanding Alzheimer's disease (AD) via the analysis of intrinsic brain connectivity [10]. Although there is converging evidence about the identity of the affected regions in fMRI, it is not clear how this abnormality affects the functional organisation of the whole brain.

Analysis tools derived from measures of network entropy have been extensively used to characterise the salient features of the structure of network systems arising in biology, physics, and the social sciences [1–3]. In particular ideas from statistical mechanics and information theory have been used to develop techniques and analyse the time evolution of network structure using analogies with both classical and quantum systems. For example, the von Neumann entropy can be used as an effective characterization of network structure, commencing from

a quantum analogue in which the Laplacian matrix plays the role of the density matrix [1]. Further development of this idea has shown the link between the von Neumann entropy and the degree statistics of pairs of nodes forming edges in a network [2], which can be efficiently computed for both directed and undirected graphs [3]. Since the eigenvalues of the density matrix reflect the energy states of a network, this approach is closely related to the heat bath analogy in statistical mechanics.

These promising approaches from statistical mechanics [4], thermodynamics [5] or quantum information [6] provide a convenient route to network characterisation. A well-explored study is an analogy of combining the networks with thermodynamic system [7]. The Hamiltonian operator identifies the energy states of a network by using the eigenvalues of a matrix characterisation. By mapping the network system occupied by a set of particles, the energy states are supported to be populated these particles in thermal equilibrium with the heat bath [7]. The occupation of particles in the energy states is populated according to the specific distribution. Specifically, these associated with the assumptions concerning the quantum spin statistics, namely Bose-Einstein and Fermi-Dirac statistics. From the relevant partition function, the thermodynamic entropy can be derived to characterise networks [7].

Although entropic network analysis using the heat bath analogy provides a useful global characterisation of network structure, they do not lend themselves to the analysis of the entropy of edge or subnetwork structure. In this paper, we explore a novel edge entropy projection which can be applied to the global network entropy computed from statistical mechanics using both the classical Boltzmann distribution and the quantum Bose-Einstein and Fermi-Dirac statistics [7]. The new characterisations of edge entropy resulting from this analysis allow us to probe in finer detail the interactions between different anatomical regions in fMRI data from healthy controls and Alzheimer's disease sufferers (AD).

It as been noted that AD subjects exhibit significantly lower regional connectivity and exhibit disrupted the global functional organisation when compared to healthy controls [8]. Because Bose-Einstein particles coalesce in low energy states and Fermi-Dirac particles have a greater tendency to occupy high energy states because of the Pauli exclusion principle, these types of spin statistics lead to very different distributions of entropy for a network with a given structure (i.e. a set of normalised Laplacian eigenvalues) [7]. Moreover, we wish to investigate them as a means of characterising differences in the network structure at low temperature. The analysis of the distribution of edge entropy within a network reveals that the different quantum statistics can be used to explore how the distribution of edge-entropy encodes the intrinsic differences in the anatomical pattern of fMRI responses between different groups having Alzheimer's disease and normal healthy controls.

This paper is organised as follows. Section 2 briefly reviews the basic concepts in network representation, especially with sophisticated study of von Neumann entropy. Section 3 reviews density matrix and Hamiltonian operator on graphs, and decompose the thermodynamic entropy on edges from Bose-Einstein and

Fermi-Dirac statistics. Section 4 provides our experimental evaluation. Finally, Sect. 5 provides the conclusion and direction for future work.

## 2 Graph Representation

In this section, we provide the basic background of graph representation and basic quantum theory. We briefly introduce the concept of the normalized Laplacian matrix as the density matrix in the definition of von Neumann entropy.

### 2.1 Preliminary

Let  $G(V, E)$  be an undirected graph with node set  $V$  and edge set  $E \subseteq V \times V$ , and let  $V$  represent the total number of nodes on graph  $G(V, E)$ . The adjacency matrix of a graph is  $A$  with the degree of node  $u$  is  $d_u = \sum_{v \in V} A_{uv}$ . Then, the Laplacian matrix is  $L = D - A$ , where  $D$  denotes the degree diagonal matrix whose elements are given by  $D(u, u) = d_u$  and zeros elsewhere. The normalized Laplacian matrix  $\tilde{L}$  of the graph  $G$  is defined as  $\tilde{L} = D^{-\frac{1}{2}} L D^{\frac{1}{2}}$ , and the spectral decomposition is  $\tilde{L} = \Phi \tilde{\Lambda} \Phi^T$ , where  $\tilde{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_{|V|})$  is the diagonal matrix with the ordered eigenvalues as elements and  $\Phi = (\varphi_1, \varphi_2, \dots, \varphi_{|V|})$  is the matrix with the ordered eigenvectors as columns.

### 2.2 von Neumann Edge Entropy

The density matrix describes a system with an ensemble of pure quantum states  $|\psi_i\rangle$  and each with probability  $p_i$ . It is defined as  $\rho = \sum_{i=1}^V p_i |\psi_i\rangle \langle \psi_i|$ . The density matrix for a graph or network can be achieved by scaling the normalised Laplacian matrix by the reciprocal of the number of nodes [1, 6]. It is defined as  $\rho = \frac{\tilde{L}}{V}$ . This interpretation opens up the possibility of characterising a graph using the von Neumann entropy from quantum information theory. Therefore, the von Neumann entropy is given in terms of the eigenvalues  $\lambda_1, \dots, \lambda_{|V|}$  of the density matrix  $\rho$ [1],

$$S_{vN} = -\text{Tr}(\rho \log \rho) = - \sum_{i=1}^{|V|} \frac{\lambda_i}{|V|} \log \frac{\lambda_i}{|V|} \tag{1}$$

In fact, Han et al. [2] have shown how to approximate the calculation of von Neumann entropy in terms of simple degree statistics. Their approximation allows the cubic complexity of computing the von Neumann entropy to be reduced to one of quadratic complexity using simple edge degree statistics, i.e.

$$S_{vN} = 1 - \frac{1}{|V|} - \frac{1}{|V|^2} \sum_{(u,v) \in E} \frac{1}{d_u d_v} \tag{2}$$

Therefore, the edge entropy decomposition is given as

$$S_{vN}^{edge}(u, v) = \frac{1}{|E|} - \frac{1}{|V||E|} - \frac{1}{|E||V|^2} \frac{1}{d_u d_v} \tag{3}$$

where  $S_{VN} = \sum_{(u,v) \in E} S_{VN}^{edge}(u,v)$ . This expression decomposes the global parameter of von Neumann entropy on each edge with the relation to the degrees from the connection of two vertexes.

### 3 Quantum Statistics and Global Entropy Decomposition

The concept of von Neumann entropy arises in the quantum domain. Here, we commence from the Hamiltonian operator in quantum statistics to develop thermodynamic entropy. We then decompose or project the global entropy onto edges using the eigenvectors of normalised Laplacian matrix.

#### 3.1 Thermodynamic Entropy

To connect the normalised Laplacian matrix to statistical mechanics and quantum statistics, we view the eigenvalues of the Laplacian matrix as the energy eigenstates of a system in contact with a heat reservoir. These determine the Hamiltonian and hence the relevant Schrödinger equation which governs the particles in the system. The particles occupy the energy states of the Hamiltonian subject to thermal agitation by the heat bath. The number of particles in each energy state is determined by the temperature, the assumed model of occupation statistics and the relevant chemical potential.

We consider the network as a thermodynamic system of  $N$  particles with energy states given by normalised Laplacian matrix  $\tilde{L}$ , which is immersed in a heat bath with temperature  $T$ . The ensemble is represented by a partition function  $Z(\beta, N)$ , where  $\beta$  is inverse of temperature  $T$ . When specified in this way, the thermodynamic entropy is given by,

$$S = k_B \left[ \frac{\partial}{\partial T} T \log Z \right]_N \tag{4}$$

with the corresponding chemical potential  $\mu$  as,

$$\mu = -k_B T \left[ \frac{\partial}{\partial N} \log Z \right]_\beta \tag{5}$$

The statistical properties of particles in the network are determined by the partition functions associated with different energy level occupation statistics. In this way, thermodynamic quantities, such as entropy, can characterise the network structure.

#### 3.2 Bose-Einstein Edge Entropy

Bose-Einstein statistics apply to indistinguishable bosons which can aggregate in the same energy state. For a system with a varying number of particles  $N$  and a chemical potential  $\mu$ , the Bose-Einstein partition function is

$$Z_{BE} = \det \left( I - e^{\beta\mu} \exp[-\beta\tilde{L}] \right)^{-1} = \prod_{i=1}^V \left( \frac{1}{1 - e^{\beta(\mu - \epsilon_i)}} \right) \tag{6}$$

From Eq. (4), the corresponding entropy is

$$\begin{aligned}
 S_{BE} = & -\text{Tr} \left\{ \log [I - e^{\beta\mu} \exp(-\beta\tilde{L})] \right\} \\
 & - \text{Tr} \left\{ \beta [I - e^{\beta\mu} \exp(-\beta\tilde{L})]^{-1} (\mu I - \tilde{L}) e^{\beta\mu} \exp(-\beta\tilde{L}) \right\}
 \end{aligned} \tag{7}$$

The entropy depends on the chemical potential for the system and hence the number of particles in the system. The equivalent density matrix for the system of particles is given by

$$\rho_{BE} = \frac{1}{\text{Tr}(\rho_1) + \text{Tr}(\rho_2)} \begin{pmatrix} \rho_1 & 0 \\ 0 & \rho_2 \end{pmatrix} \tag{8}$$

where

$$\begin{aligned}
 \rho_1 &= - \left( \exp[\beta(\tilde{L} - \mu I)] - I \right)^{-1} \\
 \rho_2 &= \left( I - \exp[-\beta(\tilde{L} - \mu I)] \right)^{-1}
 \end{aligned}$$

To compute the edge entropy projection for a system with Bose-Einstein statistics, we exploit the spectral decomposition of the normalised Laplacian matrix. The Bose-Einstein entropy can be written as

$$S_{BE}^{edge}(u, v) = \sum_{i=1}^{|V|} \sigma(\varepsilon_i) \varphi_i \varphi_i^T \tag{9}$$

where

$$\sigma_{BE}(\varepsilon_i) = - \sum_{i=1}^V \log \left( 1 - e^{\beta(\mu - \varepsilon_i)} \right) - \beta \sum_{i=1}^V \frac{(\mu - \varepsilon_i) e^{\beta(\mu - \varepsilon_i)}}{1 - e^{\beta(\mu - \varepsilon_i)}}$$

### 3.3 Fermi-Dirac Edge Entropy

Fermi-Dirac statistics apply to indistinguishable fermions with a maximum occupancy of one particle in each energy state. According to the Pauli exclusion principle, no further particles can be added to states that are already occupied. The partition function for a system subject to Fermi-Dirac occupation statistics is

$$Z_{FD} = \det \left( I + e^{\beta\mu} \exp[-\beta\tilde{L}] \right) = \prod_{i=1}^V \left( 1 + e^{\beta(\mu - \varepsilon_i)} \right) \tag{10}$$

with associated entropy given by

$$\begin{aligned}
 S_{FD} = & \text{Tr} \left\{ \log [I + e^{\beta\mu} \exp(-\beta\tilde{L})] \right\} \\
 & - \text{Tr} \left\{ \beta [I + e^{\beta\mu} \exp(-\beta\tilde{L})]^{-1} (\mu I - \tilde{L}) e^{\beta\mu} \exp(-\beta\tilde{L}) \right\}
 \end{aligned} \tag{11}$$

Similarly, the density matrix for the system is

$$\rho_{FD} = \frac{1}{\text{Tr}(\rho_3) + \text{Tr}(\rho_4)} \begin{pmatrix} \rho_3 & 0 \\ 0 & \rho_4 \end{pmatrix} \quad (12)$$

where

$$\begin{aligned} \rho_3 &= \left( I + e^{-\beta\mu} \exp[\beta\tilde{L}] \right)^{-1} \\ \rho_4 &= \left( I + e^{\beta\mu} \exp[-\beta\tilde{L}] \right)^{-1} \end{aligned}$$

Therefore, the corresponding edge entropy decomposition is,

$$S_{FD}^{edge}(u, v) = \sum_{i=1}^{|V|} \sigma(\varepsilon_i) \varphi_i \varphi_i^T \quad (13)$$

where

$$\sigma_{FD}(\varepsilon_i) = \sum_{i=1}^{|V|} \log \left( 1 + e^{\beta(\mu - \varepsilon_i)} \right) - \beta \sum_{i=1}^{|V|} \frac{(\mu - \varepsilon_i) e^{\beta(\mu - \varepsilon_i)}}{1 + e^{\beta(\mu - \varepsilon_i)}}$$

## 4 Experiments and Evaluations

In this section, we describe the application of the above methods to the analysis of interregional connectivity structure for fMRI activation networks for normal and Alzheimer's patients. We first examine the dependence of the quantum edge entropy components on node degree and temperature and compare their performance with von Neumann entropy. Then we apply edge entropy-based analysis to distinguish between different stages in the development of Alzheimer's disease, and fMRI data for normal subjects. We explore whether we can identify specific inter-regional connections and regions in the brain associated with the neuro-degeneration caused by the onset of Alzheimer's disease. To simplify the calculations, the Boltzmann constant is set to unity in our experiments.

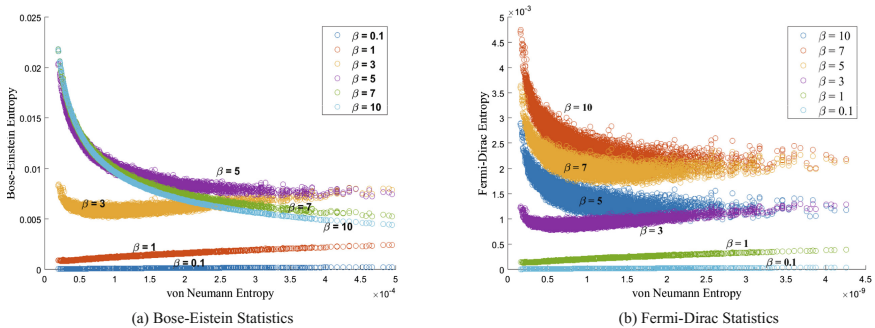
### 4.1 Dataset

The fMRI data were obtained from the ADNI initiative [9]. fMRI images of subjects brains were taken every two seconds and are used to compute the Blood-Oxygenation-Level-Dependent (BOLD) signals for different anatomical brain regions. To do this the fMRI voxels were aggregated into larger regions of interest (ROIs). The different ROIs correspond to different anatomical regions of the brain and are assigned anatomical labels to distinguish them. There are 96 such anatomical regions in each fMRI image. The correlation between the average time series in different ROIs represents the degree of functional connectivity between regions which are driven by neural activities [8].

We construct a graph to represent the pattern of activities using the cross-correlation coefficients for the average time series for pairs of ROIs. We create an undirected edge between two ROI’s if the cross-correlation co-efficient between the time series is in the top 40% of the cumulative distribution. This cross-correlation threshold is fixed over all of the available data, which provides an optimistic bias for constructing graphs. Those ROIs that have missing time series data are discarded. Subjects fall into different categories according to the degree of severity of the disease, there are normal subjects, those with early mild cognitive impairment, those with late mild cognitive impairment and those with full Alzheimer’s. The data supplied included 30 subjects with Alzheimer’s disease (AD) and 38 normal, healthy control subjects.

### 4.2 Experimental Results

We first investigate the relationship between the mean edge entropy computed using quantum statistics and von Neumann entropy. Figure 1 shows the edge entropy with varying temperatures. Both statistical entropies exhibit a transition in behaviour with respect to the von Neumann entropy with varying temperature. For example, at the high temperature ( $\beta = 0.1$ ), both quantum entropies are roughly in linear proportion to the von Neumann entropy. As the temperature reduces, they take on an approximately exponential dependence. At low temperature, the quantum edge entropies decrease monotonically with the von Neumann edge entropy ( $\beta = 10$ ). Therefore, at high temperature, the quantum and von Neumann edge entropies are proportional, while at low temperature they are in inverse proportion.

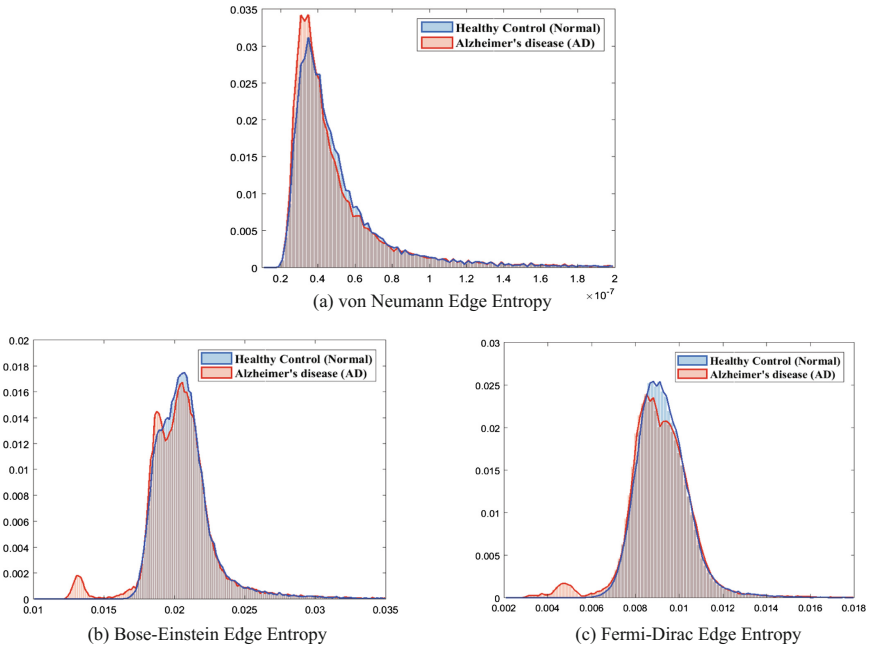


**Fig. 1.** Scatter plot of edge entropies compared to the von Neumann entropy with different value of temperatures.

However, the spread as measured by the variance of the quantum edge entropies corresponding to a fixed von Neumann entropy is also revealing. In the Bose-Einstein case, the spread of edge entropies about the mean is narrow, while in the Fermi-Dirac it exhibits a broader and more scattered pattern.

This effect is most obvious in the high-temperature region. The reason for this is that the networks possess some internal cluster or community structure. Since Bose-Einstein statistics preferentially sample the lower energy levels of the network eigenvalue spectrum, it is more susceptible to strong community structure. On the other hand, Fermi-Dirac statistics are more sensitive to a wider range of eigenvalues and are hence sensitive to both the mean and variance of the eigenvalue distribution.

We also apply the different edge entropy computations to fMRI brain networks, with the aim of determining which anatomical regions play the strongest role in the development of Alzheimer’s disease. Figure 2 the different edge entropy distribution for the Alzheimer’s disease (AD) and healthy control (Normal) samples. Compared to the von Neumann entropy which does not show a clear difference in distributions between the two groups, the quantum entropies better distinguish the detailed distribution of edge entropy. The edge entropy in the case Alzheimer’s disease tends towards lower values. This observation is more palpable in the cases of the Bose-Einstein and Fermi-Dirac edge entropy distributions, as shown in Fig. 2(b) and (c), with more edges tending to occupy the low entropy region. Moreover, the Bose-Einstein edge entropy exhibits better separation between the healthy and Alzheimer’s groups compared to that for the Fermi-Dirac distribution, since here the non-overlapping area is much larger.



**Fig. 2.** Edge entropy distribution of fMRI networks with (a) von Neumann entropy, (b) Bose-Einstein statistics and (c) Fermi-Dirac statistics. Two groups of patients, Alzheimer’s disease (AD) and healthy control (Normal).



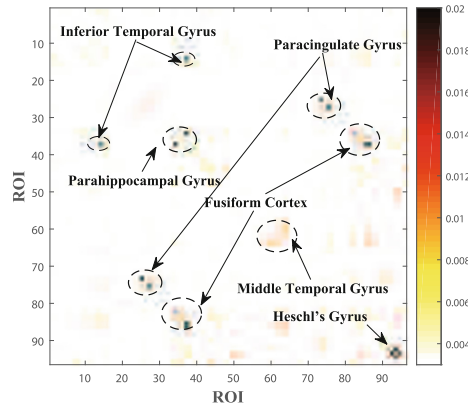
Identifying diseased regions in the brain is also important. Several studies have shown that different anatomical structures can be analysed using the properties of the corresponding ROIs, and are important for understanding brain disorders [10, 11]. Here, we use the difference in standard deviation for the quantum entropy to identify the sources of significant variance between AD and HC groups. Figure 3 plots the greatest variance of edge entropy for different anatomical regions (nodes). The entropic measurements in the brain areas, such as the Paracingulate Gyrus, Parahippocampal Gyrus, Inferior Temporal Gyrus and Temporal Fusiform Cortex, suggest that subjects with AD experience loss of interconnection between these regions in their brain network during the progression of the disease.

As listed in Table 1, the ten anatomical regions with the largest entropy differences for subjects with the full AD are Paracingulate Gyrus, Parahippocampal Gyrus, Temporal Fusiform Cortex, etc. This result is consistent with the previous study reported in [11, 12]. For example, the parahippocampal gyrus has consistently been reported as being vulnerable to pathological changes in Alzheimer's disease (AD), which is closely related to entorhinal and perirhinal subdivisions as the most heavily damaged cortical areas for the disease [13]. The Frontal Medial Cortex and Temporal Fusiform Cortex are memory-related cognitive areas. They are severely damaged by Alzheimer's disease and affect recognition memory for faces. Overall, the loss of connection between these brain regions results in significant functional impairment between healthy subjects and patients with the AD.

**Table 1.** Top 10 ROIs with the most significant difference in edge entropy between the Alzheimer's disease (AD) and Health Control (Normal) groups.

Index	ROI	ROI
1	Inferior Temporal Gyrus Left (14)	Temporal Fusiform Cortex Left (37)
2	Frontal Medial Cortex Left (25)	Frontal Medial Cortex Right (73)
3	Paracingulate Gyrus Left (27)	Paracingulate Gyrus Right (75)
4	Parahippocampal Gyrus Left (34)	Temporal Fusiform Cortex Left (37)
5	Parahippocampal Gyrus Left (34)	Parahippocampal Gyrus Right (82)
6	Temporal Fusiform Cortex Left (37)	Temporal Fusiform Cortex Right (85)
7	Temporal Fusiform Cortex Left (37)	Temporal Fusiform Cortex Right (86)
8	Inferior Temporal Gyrus Right (63)	Temporal Fusiform Cortex Right (86)
9	Planum Polare Right (92)	Heschl's Gyrus Right (93)
10	Heschl's Gyrus Right (93)	Planum Temporale Right (94)

In conclusion, both statistical methods and von Neumann edge entropies can be used to represent changes in network structure. Compared to the von Neumann edge entropy, quantum edge entropies are more sensitive to sample variance associated with the degree distribution. At high-temperature region, the quantum statistics have similar degree sensitivity. However, at low-temperature,



**Fig. 3.** Significant differences between edge entropy associated with diseased areas in the brain. We use the standard deviation of quantum entropy to identify the divergence between AD and HC groups for each edge.

Bose-Einstein statistics reflect strong community structure while Fermi-Dirac statistics are more suitable for representing a detailed structure of the degree distribution.

## 5 Conclusion

In this paper, we show how to decompose the global network entropies resulting from quantum occupation statistics onto the constituent edges of a graph. We refer to the resulting quantum statistical quantities as Bose-Einstein and Fermi-Dirac edge-entropies. The method uses the normalised Laplacian matrix as the Hamiltonian operator of the network to compute the corresponding partition functions. We undertake experiments to analyse the quantum edge entropies and compare them to their von Neumann counterparts. Experiments reveal that both the Bose-Einstein and Fermi-Dirac edge entropy distributions can effectively in characterising detailed variations in the network structure. They both outperform the von Neumann entropy in this respect. Finally, we apply this novel method to provide insights into the neuropathology of Alzheimer's disease. The quantum edge entropy distribution is capable of discriminating between subjects suffering from Alzheimer's and healthy subjects.

## References

1. Passerini, F., Severini, S.: The von Neumann entropy of networks. *Int. J. Agent Technol. Syst.* **1**, 5867 (2008)
2. Han, L., Escolano, F., Hancock, E.R., Wilson, R.C.: Graph characterizations from von Neumann entropy. *Pattern Recogn. Lett.* **33**, 19581967 (2012)

3. Ye, C., Wilson, R.C., Comin, C.H., Costa, L.D.F., Hancock, E.R.: Approximate von Neumann entropy for directed graphs. *Phys. Rev. E* **89**(5), 052804 (2014)
4. Park, J., Newman, M.: Statistical mechanics of networks. *Phys. Rev. E* **70**(6), 066117 (2004)
5. Estrada, E., Hatano, N.: Communicability in complex networks. *Phys. Rev. E* **77**, 036111 (2008)
6. Anand, K., Bianconi, G., Severini, S.: Shannon and von Neumann entropy of random networks with heterogeneous expected degree. *Phys. Rev. E* **83**(3), 036109 (2011)
7. Wang, J., Wilson, R.C., Hancock, E.R.: Spin statistics, partition functions and network entropy. *J. Complex Netw.* **5**(6), 858883 (2017)
8. Wang, J., Wilson, R.C., Hancock, E.R.: Detecting Alzheimer's disease using directed graphs. In: Foggia, P., Liu, C.-L., Vento, M. (eds.) *GbrPR 2017. LNCS*, vol. 10310, pp. 94–104. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-58961-9\\_9](https://doi.org/10.1007/978-3-319-58961-9_9)
9. Petersen, R.C., Aisen, P.S., Beckett, L.A., et al.: Alzheimers disease neuroimaging initiative (ADNI): clinical characterization. *Neurology* **74**(3), 201–209 (2010)
10. Rubinov, M., Sporns, O.: Complex network measures of brain connectivity: uses and interpretations. *Neuroimage* **52**(3), 1059–1069 (2010)
11. Rombouts, S.A., Barkhof, F., Goekoop, R., Stam, C.J., Scheltens, P.: Altered resting state networks in mild cognitive impairment and mild Alzheimer's disease: an fMRI study. *Hum. Brain Mapp.* **26**(4), 231–239 (2005)
12. Khazaei, A., Ebrahimzadeh, A., Babajani-Ferem, A.: Classification of patients with MCI and AD from healthy controls using directed graph measures of resting-state fMRI. *Behav. Brain Res.* **322**, 339–350 (2016). ISSN 0166-4328
13. Van Hoesen, G.W., Augustinack, J.C., Dierking, J., Redman, S.J., Thangavel, R.: The parahippocampal gyrus in Alzheimer's disease: clinical and preclinical neuroanatomical correlates. *Ann. New York Acad. Sci.* **911**(1), 254–274 (2000)



# Approximating GED Using a Stochastic Generator and Multistart IPFP

Nicolas Boria<sup>(✉)</sup>, Sébastien Bougleux, and Luc Brun

Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, Caen, France  
{boria,luc.brun}@ensicaen.fr, bougleux@unicaen.fr

**Abstract.** The Graph Edit Distance defines the minimal cost of a sequence of elementary operations transforming a graph into another graph. This versatile concept with an intuitive interpretation is a fundamental tool in structural pattern recognition. However, the exact computation of the Graph Edit Distance is  $\mathcal{NP}$ -complete. Iterative algorithms such as the ones based on Franck-Wolfe method provide a good approximation of true edit distance with low execution times. However, underlying cost function to optimize being neither concave nor convex, the accuracy of such algorithms highly depends on the initialization. In this paper, we propose a smart random initializer using promising parts of previously computed solutions.

**Keywords:** Graph edit distance · Parallel gradient descents  
Multistart · Stochastic warm start

## 1 Introduction

Computing a similarity or a dissimilarity measure between graphs is a major challenge in pattern recognition. One of the most well-known and used approach to compute a distance between two graphs is the Graph Edit Distance (GED) [12]. Computing the GED consists in finding a sequence of graph edit operations (insertions, deletions and substitutions of vertices and edges) that transforms a graph into another with a minimal cost. Such a sequence of edit operations is called an edit path, and the edit distance between two graphs  $G$  and  $H$  is defined by  $GED(G, H) = \min_{\gamma \in \Gamma(G, H)} \sum_{e \in \gamma} c(e)$ , where  $\Gamma(G, H)$  denotes the set of edit paths between  $G$  and  $H$  and  $c(e)$  denotes the cost of an elementary operation  $e$  belonging to the edit path  $\gamma$ .

If both graphs are simple and if the cost between vertices and edges remains fixed, one can show [3] that the edit distance between two graphs  $G$  and  $H$  of respective orders  $n$  and  $m$  may be formulated as the following quadratic problem:  $GED(G, H) = \min_{x \in \Pi_{n,m}} \frac{1}{2} x^t \Delta x + c^t x$ , where  $\Pi_{n,m}$  denotes the set of vectorized assignment matrices between  $V_G$  and  $V_H$ . Such a matrix  $x$  encodes for each element of  $V_G$  one and only one operation (either substitution or deletion).

---

Work supported by Region Normandie under project RIN AGAC.

In the same way,  $x$  encodes for each element of  $V_H$  either a substitution or an insertion. The matrix  $\Delta$  encodes the cost of edge operations while  $c$  encodes the cost of operations on vertices. Computing the GED is NP-hard, so several heuristics were proposed to compute approximate solutions in polynomial time.

The design of approximate solutions to the GED problem has been strongly stimulated by the introduction of an approximation of the GED problem into a Linear Sum Assignment Problem with Edition (LSAPE) [9]. This approximation consists in associating to each node of two graphs  $G$  and  $H$  a substructure and to populate a cost matrix encoding: the costs of matching two substructures, the cost of inserting one substructure in  $H$  and the cost of removing one substructure from  $G$ . Given such a cost matrix  $\tilde{c}$ , the assignment matrix  $x$  minimizing  $\tilde{c}^t x$  provides a set of elementary operations on vertices from which an edit path may be deduced. The cost of this edit path provides an upper bound for the graph edit distance. This minimization step may be solved in polynomial time. This transformation of an  $\mathcal{NP}$ -complete problem into a minimization problem with a polynomial complexity is the major advantage of the LSAPE approximation. However the computation of the cost matrix  $\tilde{c}$  may require non polynomial execution times. Different types of substructures have been defined in [4, 8, 9, 14].

However, LSAPE is based on a linear approximation of a quadratic problem. In order to get a finer approximation of the graph edit distance, several methods use variants of local search such as simulated annealing in order to improve an initial estimation of the edit distance [6, 10, 11]. A slightly different approach [3], consists in using the Franck Wolfe minimization scheme [7] from an initial guess. This algorithm converges by iterations towards a local minimum of the quadratic function usually close from the initial guess. This heuristic provides close approximations of the graph edit distance on small graphs but is sensitive to the solution used to initialize the method. An heuristic to reduce the influence of the initial guess has been proposed in [5]. This heuristic is based on the use of multiple initial guesses either deduced from a set of solutions of the LSAPE problem or based on the generation of random assignment matrices. The common drawback of these two heuristics is that the generation of initial solutions does not take into account information provided by runs of Franck-Wolfe method which may have converged.

In this paper we propose a new heuristic based on alternative runs of the generation of initial solutions and the determination of their associated local minima using Franck-Wolfe method. This method is described in Sect. 3. The proposed method is evaluated in Sect. 4 through several experiments.

## 2 Preliminaries

Throughout the paper, we will use the same concepts and notations as those introduced in [3]. Vertices of graphs  $G$  and  $H$  are numbered respectively from 1 to  $n$  and from 1 to  $m$ , and two virtual vertices indexed as  $n + 1$  in  $G$  and as  $m + 1$  in  $H$  are added. These virtual vertices, denoted by  $\epsilon_G$  and  $\epsilon_H$ , correspond respectively to insertions and deletions. An assignment  $i \rightarrow \epsilon_H$  (resp.  $\epsilon_G \rightarrow j$ ) corresponds to the deletion of vertex  $i$  of  $G$  (resp. insertion of vertex  $j$  of  $H$ ).

```

1 begin
2   Minimize a linear approximation of  $Q$  around the current solution  $\mathbf{x}$  in the
       discrete domain by solving an LSAP:  $\mathbf{b}^* \leftarrow \underset{\mathbf{b} \in \pi_{n,m,\epsilon}}{\operatorname{argmin}} (\mathbf{x}^\top \Delta) \mathbf{b}$ 
3   Perform the descent by minimizing  $Q$  along the segment  $[\mathbf{x}, \mathbf{b}]$  in the
       continuous domain:  $\alpha^* \leftarrow \underset{\alpha \in [0,1]}{\operatorname{argmin}} Q(\mathbf{x} + \alpha(\mathbf{b}^* - \mathbf{x}))$ 
4   Update  $x : \mathbf{x} \leftarrow \mathbf{x} + \alpha^*(\mathbf{b}^* - \mathbf{x})$ 
5   Repeat steps 2 to 4 until  $\mathbf{x}^\top \Delta(\mathbf{x} - \mathbf{b}^*) < \beta |Q(\mathbf{x}) + \mathbf{x}^\top \Delta(\mathbf{b}^* - \mathbf{x})|$  holds for
       a given scalar  $\beta \in (0, 1)$ , or if a given number of iterations is reached

```

**Algorithm 1.** FW( $\Delta, \mathbf{x}$ )

In this context, a solution for GED will be described as an *error-correcting assignment matrix*  $x$ , where all vertices of  $G$  are assigned to a single element of  $H \cup \{\epsilon_H\}$ , and all vertices of  $H$  are assigned to a single element of  $G \cup \{\epsilon_G\}$ . The polytope  $\Pi_{n,m}^\epsilon$  of *error-correcting assignment matrices* thus contains all matrices of dimensions  $(n + 1) \times (m + 1)$ , with binary values, and with a single 1 in each line and in each column, except for the last line and the last column. We naturally extend the concept to matrices with fractional values: we call *error-correcting bistochastic matrix* any matrix where the sum of all cells for each line except the last one and each column except the last one amounts exactly to 1.

Given a cost matrix  $\Delta$  and an initial continuous or discrete candidate solution  $x$ , Algorithm 1 describes Frank-Wolfe algorithm FW( $\Delta, \mathbf{x}$ ). In the following, we denote by IPFP the method that consists in running FW( $\Delta, \mathbf{x}$ ) and projecting the returned solution in the discrete space. See [3] for more details.

### 3 RANDPOST Algorithm

The conception of algorithm RANDPOST finds its origin in the following intuition regarding the - often conflicting - criteria that a smart initial solution generator should fulfill. On the one hand, a smart generator should propose solutions that are well distributed inside the  $\Pi_{n,m}^\epsilon$  polytope, in order to increase the diversity among local minima that are ultimately returned. We call this criterium *exploration criterium*. On the other hand, we obviously wish that one of the initial solutions ultimately led to a global minimum, so that a good generator should somehow already generate solutions that includes “smart” assignments. We call this criterium *quality criterium*.

Building up on the progresses of the multistart IPFP (mIPFP) algorithm [5], we propose a new algorithm that explores the polytope and generates new solutions by taking advantage of the whole statistical information contained in the set of solutions returned by mIPFP. This algorithm is presented in Sect. 3.2. We also devise a new parameterized random generator that is described in Sect. 3.1.

### 3.1 Generating Initial Solutions with Parameterized Number of Insertions and Deletions

With respect to the initial random generator used in [5], where vertices were randomly assigned by the use of the `std::random_shuffle` procedure of the C++ standard library, resulting in a random proportion of insertions and deletions, we decided to use a random generator with a parameterized proportion of insertions and deletions.

We focus our analysis on the number of deletions, as - for given  $n = |G|$  and  $m = |H|$  - the number of deletions is completely determined by it, namely,  $\#insertions \equiv \#deletions - n + m$ .

Namely, given a parameter  $\alpha \in (0, 1)$ , we used a new initial random generator `RANDGEN( $\alpha$ )` which generates solutions with an expected number of deletions equal to  $\alpha n + (1 - \alpha) \max\{(n - m), 0\}$ . In other words,  $\alpha$  denotes the expected proportion of “unnecessary” deletions of vertices in the set of initial solutions, reminding that if  $n > m$  any feasible assignment will have to assign at least  $n - m$  vertices to the virtual node  $\varepsilon_H$  which thus correspond “necessary” to deletions.

### 3.2 Stochastic Generation of New Initial Solutions Based on Several Refined Solutions

We describe here the functioning of `RANDPOST` (Algorithm 2). Given a pair of graphs  $G$  and  $H$ , the algorithm starts by running `mIPFP`, which outputs a set  $S^*$  of  $r$  solutions for the problem. Each of these  $r$  solutions is represented as an error-correcting assignment matrix  $x$ . A matrix  $\Psi$  is then created by simply summing all of these  $r$  matrices and dividing all values by  $r$ . Hence,  $\forall i \in [1, n + 1], j \in [1, m + 1]$ ,  $\Psi_{i,j}$  represents the proportion of solutions where  $i$  is assigned to  $j$  among the  $r$  solutions of  $S^*$ . Matrix  $\Psi$  (which is an error-correcting bistochastic matrix) is then used as a probability distribution to compute a new set of  $k$  solutions which are subsequently refined using `IPFP`, and the first  $r$  that have converged among these  $k$  parallel processes are added to the set  $S^*$ . The whole

```

1 begin
2   Generate an initial set  $S \subset \mathcal{D}_{n,m,\varepsilon}$  of  $k$  solutions using RANDGEN( $\alpha$ )
3   Start refining all solutions in  $S$  using IPFP, and stop the refinement process
   when  $r$  of them have converged, which results in a set of  $r$  improved
   solutions  $S^*$ 
4   for  $i=1$  to  $l$  do
5     Update matrix  $\Psi$  in the following way:  $\Psi \leftarrow \sum_{x \in S^*} x / |S^*|$ .
6     Generate a new set  $S$  of  $k$  solutions using a random generator that uses
   the probability distribution described by Eq. (1).
7     Start refining all solutions in  $S$  using IPFP, and add the  $r$  first to have
   converged to  $S^*$ 

```

**Algorithm 2.** `RANDPOST( $k, r, l$ )`

sequence that updates  $\Psi$ , generates new solutions and finally refines them is repeated  $l$  times.

Parameters  $k$  and  $r$  enables to speed up the algorithm when  $k \gg r$ : some of the  $k$  initial solutions might require many iterations of IPFP in order to converge to a local minimum, so that by launching  $k$  parallel IPFPs and stopping all them when  $r$  of them have converged, the whole process is likely to be faster than launching  $r$  parallel IPFPs and waiting for all of them to converge.

The intuitive idea behind algorithm RANDPOST is the following: if an assignment  $i \rightarrow j$  appears with a high frequency within solutions of the set  $S^*$ , then this assignment is likely to be part of many good solutions for the problem at hand. Hence, the algorithm generates  $k$  new solutions in a stochastic way, where the probability for a given assignment to be part of a solution is higher for assignments with high  $\Psi_{i,j}$  value, and thus high frequency.

To be even more precise, the random generator assigns each vertex of  $G$  to a vertex of  $H \cup \{\epsilon_H\}$  following a greedy procedure. The matrix  $x$  of dimensions  $(n + 1) \times (m + 1)$  (which will eventually contain the solution returned by the algorithm) is initialized with zeros, and whenever an assignment  $i \rightarrow j$  is made by the algorithm, this translates to  $x_{i,j} \leftarrow 1$ . At the end of the algorithm  $x$  should be an error-correcting permutation. The random generator assigns iteratively vertices from 1 to  $n$  based on the following probabilistic distribution  $P_i$ , where  $P_i(j)$  denotes the probability for vertex  $i$  of  $G$  to be assigned to vertex  $j$  of  $H$  by the generator, given a partial assignment of vertices from 1 to  $i - 1$ :

$$\begin{aligned}
 &\forall j = 1, \dots, m + 1, \quad P_1(j) = \Psi(j) \\
 &\forall i = 2, \dots, n, \forall j = 1, \dots, m + 1 \\
 P_i(j) = &\begin{cases} 0 & \text{if } j \neq m + 1 \text{ and } \exists h < i, \text{ s.t. } x_{h,j} = 1 \\ \frac{\Psi_{i,j}}{1 - \sum_{h=1}^m \left( \sum_{l=1}^{i-1} x_{h,l} \Psi_{i,l} \right)} & \text{otherwise} \end{cases}
 \end{aligned} \tag{1}$$

Finally, all vertices of  $H$  that have been left unassigned at the end of the procedure are all assigned to  $\epsilon_G$ . First let us prove that matrix  $x$  produced by the proposed random generator is always an error-correcting permutation matrix. This is done by putting together the two following facts: (1) by construction, each line of  $x$  but the last one has exactly one value set to 1 and all the others set to 0 (a single assignment is made for a single line at each step of the generator), (2) the probability distribution described by (1) ensures that no vertex  $j$  of  $H$  is assigned twice (once assigned, its probability of being assigned again is zero), and the very last step ensures that each one is assigned at least once.

Let us briefly prove that (1) defines a proper probability distribution. It is easy to verify that  $\sum_{j=1}^{m+1} P_1(j) = 1$  by simply reminding that  $\Psi$  is an error-correcting bistochastic matrix. For  $i = 2, \dots, n$ , consider a matrix  $\tilde{\Psi}$  which values are as follows:

$$\tilde{\Psi}_{i,j} = \begin{cases} 0 & \text{if } j \neq m + 1 \text{ and } \exists h < i, \text{ s.t. } x_{h,j} = 1 \\ \Psi_{i,j} & \text{otherwise} \end{cases}$$



It is easy to verify that:

$$\forall i = 2, \dots, n \quad \sum_{j=1}^{m+1} \tilde{\Psi}_{i,j} = 1 - \sum_{h=1}^m \left( \sum_{l=1}^{i-1} x_{h,l} \Psi_{i,l} \right) \quad (2)$$

We finally prove that (1) defines a proper probability distribution:

$$\forall i = 2, \dots, n \quad \sum_{j=1}^{m+1} P(i \rightarrow j) = \frac{\sum_{j=1}^{m+1} \tilde{\Psi}_{i,j}}{1 - \sum_{h=1}^m \left( \sum_{l=1}^{i-1} x_{h,l} \Psi_{i,l} \right)} \stackrel{(2)}{=} 1$$

Finally, whenever the stochastic generator produces a candidate solution that has already been generated earlier, the solution is discarded and a new solution is produced using a slightly flatter (and thus more explorative) distribution.

## 4 Experimental Results

In this section, we evaluate the proposed method through several experiments, in order to determine as clearly as possible the relevance and importance of the *exploration* and *quality* criteria described in Sect. 3.

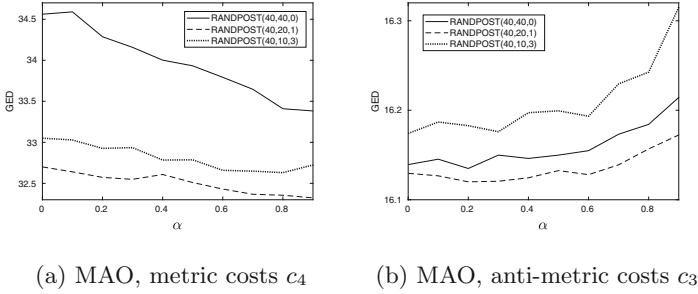
### 4.1 Datasets and Protocol

Table 1 presents the chemical datasets that were used in our experiments. MAO, PAH, MUTA10-70 and MUTAmix were considered in ICPR 2016 – Graph Distance Contest [2]. We also extracted 25 graphs from ClinTox [13], and 10 graphs with more than 100 vertices from MUTA.

**Table 1.** Characteristics of datasets

Dataset	#graphs	Avg order	Labels on nodes/edges
MAO	68	18.4	Labeled
PAH	94	20.7	Unlabeled
MUTA10-70	10	10-70	Labeled
MUTAmix	10	45	Labeled
MUTA100+	10	131.6	Labeled
ClinTox	25	115.7	Labeled

We evaluated the four following versions of  $\text{RANDPOST}(k, r, l)$ :  $\text{RANDPOST}(40, 40, 0)$ ,  $\text{RANDPOST}(40, 20, 1)$ ,  $\text{RANDPOST}(40, 10, 3)$  and  $\text{RANDPOST}(40, 5, 7)$ . This choice of parameters is determined by the following idea: considering two algorithms  $\text{RANDPOST}(k, r_1, l_1)$  and  $\text{RANDPOST}(k, r_2, l_2)$  such that  $r_1(l_1 + 1) = r_2(l_2 + 1)$  and  $r_1 > r_2$ , their relative performances can be compared without bias as the



**Fig. 1.** Behavior of RANDPOST w.r.t. parameter  $\alpha$ , metric vs. anti-metric costs

overall number of candidate solutions is the same (in our case, all four algorithms generate exactly 40 candidates), while the latter algorithm performs better on the quality criteria, and the former on the exploration one. We thus consider that  $r$  represents the exploration parameter, while  $l$  represents the quality one.

Regarding cost functions, we tested all algorithms with four different sets of costs:  $c_1$ ,  $c_2$  and  $c_3$  correspond to the costs used in [2], while  $c_4$  is the cost function used in [5] and references therein. Note that  $c_1, c_2$  and  $c_4$  correspond to metric costs where a substitution cost of two elements is lower or equal to the cost of the removal of the first element together with the insertion of the second one. Conversely,  $c_3$  is an anti-metric cost violating this last inequality. The main idea between these two classes of cost functions is that metric cost functions favor substitutions while the anti-metric ones favor deletions and insertions.

All tests were performed using 4 AMD Opteron processors at 2.6 Ghz with 512G of RAM. The number of parallel threads was limited to 40 (which corresponds to parameter  $k$ ). The code for the algorithm is written in C++.

### 4.2 Behavior of the Algorithm w.r.t. Parameter $\alpha$

We tested three versions of RANDPOST with several values for parameter  $\alpha$  of RANDGEN, and several cost functions (see the previous section). The most significant results are presented in Fig. 1 for MAO, a dataset with enough and relatively simple instances so that interesting statistical tendencies can emerge. Contrasting tendencies can be observed with the metric cost function  $c_4$  and the anti-metric one  $c_3$ . Interestingly, the algorithm performs better as the initial proportion of “unfavored” choice rises. We believe that this is due to the design of the IPFP gradient descent, which is likely to find a better local minimum when starting from a solution including a greater number of “neutral” (in the sense of easily improvable) assignments.

Unfortunately, the behavior that we observe on MAO does not emerge with the same clarity on more complex datasets containing bigger or unlabeled graphs. However, it seems that IPFP requires a medium value for  $\alpha$  (around 0.4) to perform best when dealing with unlabeled graphs. For bigger graphs (more than 40 vertices) high values of  $\alpha$  seem to produce better starting points for IPFP, independently of cost functions.

**Table 2.** Experimental results of  $\text{RANDPOST}(k, r, l)$ , cost  $c1$

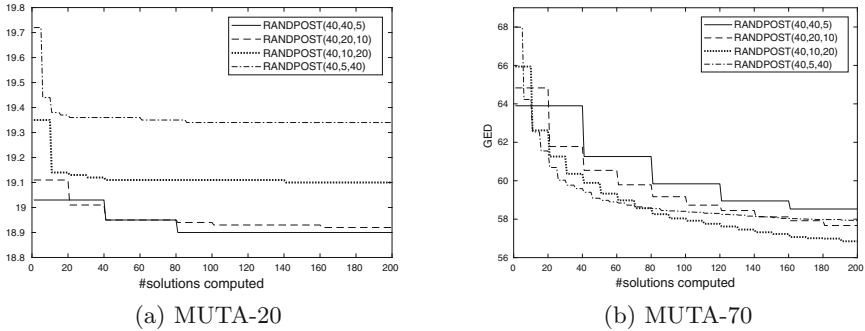
Algorithms	MAO $\alpha = 0$				PAH $\alpha = 0.3$				ClinTox $\alpha = 0.9$			
	Time	GED	err.	%best	Time	GED	err.	%best	Time	GED	err.	%best
RANDPOST(40, 1, 0)	0.013	34.43	10.30	25	0.013	36.94	24.82	1	3.542	209.42	52.12	0
RANDPOST(40, 40, 0)	0.074	24.16	0.03	98	0.099	21.23	9.11	19	17.205	167.76	10.46	2
RANDPOST(40, 20, 1)	0.029	<b>24.14</b>	<b>0.01</b>	<b>100</b>	0.038	20.71	8.59	27	13.330	163.18	5.88	10
RANDPOST(40, 10, 3)	0.051	24.19	0.06	98	0.063	<b>20.42</b>	<b>8.30</b>	<b>33</b>	19.514	160.24	2.94	30
RANDPOST(40, 5, 7)	0.144	24.48	0.35	89	0.116	20.90	8.78	26	29.278	<b>157.98</b>	<b>0.69</b>	<b>76</b>
Algorithms	MUTA 10 $\alpha = 0$				MUTA 20 $\alpha = 0.2$				MUTA 30 $\alpha = 0.8$			
	Time	GED	err.	%best	Time	GED	err.	%best	Time	GED	err.	%best
RANDPOST(40, 1, 0)	0.013	13.19	1.21	60	0.012	33.35	14.49	23	0.027	73.80	49.51	5
RANDPOST(40, 40, 0)	0.020	<b>11.98</b>	<b>0.00</b>	<b>100</b>	0.080	19.00	0.14	86	0.235	25.68	1.39	42
RANDPOST(40, 20, 1)	0.028	<b>11.98</b>	<b>0.00</b>	<b>100</b>	0.041	<b>18.96</b>	<b>0.10</b>	<b>91</b>	0.128	25.28	0.99	51
RANDPOST(40, 10, 3)	0.062	<b>11.98</b>	<b>0.00</b>	<b>100</b>	0.062	19.03	0.17	89	0.181	<b>25.07</b>	<b>0.78</b>	<b>61</b>
RANDPOST(40, 5, 7)	0.148	12.01	0.03	97	0.153	19.33	0.47	73	0.452	25.51	1.22	51
Algorithms	MUTA 40 $\alpha = 0.6$				MUTA 50 $\alpha = 0.9$				MUTA 60 $\alpha = 0.9$			
	Time	GED	err.	%best	Time	GED	err.	%best	Time	GED	err.	%best
RANDPOST(40, 1, 0)	0.063	83.94	50.23	2	0.123	81.67	44.83	5	0.246	98.55	51.97	5
RANDPOST(40, 40, 0)	0.575	36.07	2.36	26	1.141	40.10	3.26	20	2.120	50.64	4.06	11
RANDPOST(40, 20, 1)	0.302	35.00	1.29	46	0.565	38.56	1.72	31	1.158	48.95	2.37	24
RANDPOST(40, 10, 3)	0.391	<b>34.31</b>	<b>0.60</b>	<b>67</b>	0.886	<b>37.57</b>	<b>0.73</b>	<b>61</b>	1.862	47.69	1.11	54
RANDPOST(40, 5, 7)	0.516	34.85	1.14	53	1.465	37.84	1.00	55	3.133	<b>47.33</b>	<b>0.75</b>	<b>56</b>
Algorithms	MUTA 70 $\alpha = 0.9$				MUTA 100+ $\alpha = 0.9$				MUTAmix $\alpha = 0.9$			
	Time	GED	err.	%best	Time	GED	err.	%best	Time	GED	err.	%best
RANDPOST(40, 1, 0)	0.528	84.18	25.80	6	3.181	259.28	37.88	0	0.111	155.71	21.68	6
RANDPOST(40, 40, 0)	3.641	63.90	5.52	12	19.67	234.24	12.84	1	0.848	136.08	2.05	42
RANDPOST(40, 20, 1)	2.559	61.45	3.07	25	12.39	227.49	6.09	9	0.455	135.16	1.13	57
RANDPOST(40, 10, 3)	3.573	59.93	1.55	49	18.51	224.50	3.10	34	0.634	<b>134.75</b>	<b>0.72</b>	<b>68</b>
RANDPOST(40, 5, 7)	8.181	<b>59.44</b>	<b>1.06</b>	<b>56</b>	28.70	<b>222.15</b>	<b>0.75</b>	<b>78</b>	1.117	135.06	1.03	55

### 4.3 Performance of RANDPOST

Table 2 presents the performance of the four versions of  $\text{RANDPOST}(k, l)$  that we mentioned earlier, plus  $\text{RANDPOST}(1, 0)$  which corresponds to a single run of IPFP starting from a random candidate solution. For each pair of graphs in each dataset, we extracted the best known GED among those returned by a set of 14 algorithms (9 algorithms of [2] + 5 versions of  $\text{RANDPOST}$ ), except for ClinTox and MUTA100+ that weren't part of the benchmark in [1]. For these two datasets, the best GED was extracted from our 5 algorithms alone. The "err." column presents the mean error w.r.t. best known solutions, while the "%best" column presents the proportion of pairs of graphs for which the best known GED was found. For each dataset, we selected the value of  $\alpha$  leading to a minimal mean GED over all 5 tested algorithms. The selected value is indicated in the table. Due to space restrictions, we present results regarding the metric cost  $c1$  only. The same tendencies can be observed with all the other cost functions.

The tendencies that emerge from Table 2 are quite clear: the more qualitative versions of  $\text{RANDPOST}(k, r, l)$  perform better than all algorithms presented in [2] on datasets with labeled graphs containing at least 60 vertices.

Under this threshold, the balance between exploration and quality criteria that performs better GED favors more exploratory methods as the size of the graphs decreases. Further analysis shows that the phenomenon is deeply linked to the speed and quality of convergence of the algorithms: a more exploratory version of RANDPOST will ultimately converge to better GED estimations, but it will also converge at a slower rate. On the other hand, bigger graphs lead to slower overall convergence rates. These two phenomena are visible in Fig. 2. Both plots represent the improvement in GED estimations over the successive loops of RANDPOST. Each stairstep measures the best GED computed in a loop, and as the  $x$ -axis represents the number of computed solution, the length of the steps equals  $r$  for each algorithm  $RANDPOST(k, r, l)$ .



**Fig. 2.** Convergence of RANDPOST on datasets MUTA-20 and MUTA-70

When dealing with smaller graphs, qualitative methods converge very rapidly to suboptimal solutions, while the exploratory ones converge more slowly to better GED estimations. On the other hand, when dealing with bigger graphs, the fast rate of convergence of qualitative methods becomes a strength rather than a flaw, Fig. 2b shows that when the number of computed solutions is limited to 40 (which corresponds to the results in Table 2), none of the algorithms has yet converged, so that the faster converging algorithm yields better results. This phenomenon eventually reverses on the long run: as an example, Fig. 2b suggests that the limit on the number of computed solutions must be brought as high as 90 for  $RANDPOST(40, 10, 20)$  to outperform  $RANDPOST(40, 5, 40)$  on MUTA70.

## 5 Conclusion

Using a new iterative IPFP-based algorithm relying on stochastically generated solutions, we investigated the relative importance of exploration and quality criteria when generating candidate solutions for a multistart version of IPFP. Our results suggest that the balance leading to better GED estimations depends mostly on some ratio between the dimension of the problem at hand and the overall number of generated solutions.

## References

1. Abu-Aisheh, Z., Raveaux, R., Ramel, J.-Y.: A graph database repository and performance evaluation metrics for graph edit distance. In: Liu, C.-L., Luo, B., Kropatsch, W.G., Cheng, J. (eds.) GbRPR 2015. LNCS, vol. 9069, pp. 138–147. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-18224-7\\_14](https://doi.org/10.1007/978-3-319-18224-7_14)
2. Abu-Aisheh, Z., et al.: Graph edit distance contest: results and future challenges. *Pattern Recogn. Lett.* **100**, 96–103 (2017). <https://doi.org/10.1016/j.patrec.2017.10.007>
3. Bougleux, S., Brun, L., Carletti, V., Foggia, P., Gaüzère, B., Vento, M.: Graph edit distance as a quadratic assignment problem. *Pattern Recogn. Lett.* **87**, 38–46 (2017). <https://doi.org/10.1016/j.patrec.2016.10.001>
4. Carletti, V., Gaüzère, B., Brun, L., Vento, M.: Approximate graph edit distance computation combining bipartite matching and exact neighborhood substructure distance. In: Liu, C.-L., Luo, B., Kropatsch, W.G., Cheng, J. (eds.) GbRPR 2015. LNCS, vol. 9069, pp. 188–197. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-18224-7\\_19](https://doi.org/10.1007/978-3-319-18224-7_19)
5. Daller, E., Bougleux, S., Gaüzère, B., Brun, L.: Approximate graph edit distance from several assignments and multiple IPFP. In: International Conference on Pattern Recognition Applications and Methods (2018). <https://doi.org/10.5220/0006599901490158>
6. Ferrer, M., Serratos, F., Riesen, K.: A first step towards exact graph edit distance using bipartite graph matching. In: Liu, C.-L., Luo, B., Kropatsch, W.G., Cheng, J. (eds.) GbRPR 2015. LNCS, vol. 9069, pp. 77–86. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-18224-7\\_8](https://doi.org/10.1007/978-3-319-18224-7_8)
7. Frank, M., Wolfe, P.: An algorithm for quadratic programming. *Nav. Res. Logist. Q.* **3**(1–2), 95–110 (1956)
8. Gaüzère, B., Bougleux, S., Brun, L.: Approximating graph edit distance using GNCCP. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) S+SSPR 2016. LNCS, vol. 10029, pp. 496–506. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_44](https://doi.org/10.1007/978-3-319-49055-7_44)
9. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**, 950–959 (2009). <https://doi.org/10.1016/j.imavis.2008.04.004>
10. Riesen, K., Bunke, H.: Improving bipartite graph edit distance approximation using various search strategies. *Pattern Recogn.* **48**(4), 1349–1363 (2015). <https://doi.org/10.1016/j.patcog.2014.11.002>
11. Riesen, K., Fischer, A., Bunke, H.: Improved graph edit distance approximation with simulated annealing. In: Foggia, P., Liu, C.-L., Vento, M. (eds.) GbRPR 2017. LNCS, vol. 10310, pp. 222–231. Springer, Cham (2017). [https://doi.org/10.1007/978-3-319-58961-9\\_20](https://doi.org/10.1007/978-3-319-58961-9_20)
12. Sanfeliu, A., Fu, K.S.: A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man Cybern.* **13**(3), 353–362 (1983). <https://doi.org/10.1109/TSMC.1983.6313167>
13. Wu, Z., et al.: MoleculeNet: a benchmark for molecular machine learning. *Chem. Sci.* **9**, 513–530 (2018). <https://doi.org/10.1039/C7SC02664A>
14. Zeng, Z., Tung, A.K.H., Wang, J., Feng, J., Zhou, L.: Comparing stars: on approximating graph edit distance. *Proc. VLDB Endow.* **2**(1), 25–36 (2009). <https://doi.org/10.14778/1687627.1687631>



# Offline Signature Verification by Combining Graph Edit Distance and Triplet Networks

Paul Maergner<sup>1</sup>(✉), Vinaychandran Pondenkandath<sup>1</sup>, Michele Alberti<sup>1</sup>,  
Marcus Liwicki<sup>1</sup>, Kaspar Riesen<sup>2</sup>, Rolf Ingold<sup>1</sup>, and Andreas Fischer<sup>1,3</sup>

<sup>1</sup> DIVA Group, University of Fribourg, 1700 Fribourg, Switzerland  
{paul.maergner,vinaychandran.pondenkandath,michele.alberti,  
marcus.liwicki,rolf.ingold,andreas.fischer}@unifr.ch

<sup>2</sup> Institute for Information Systems, University of Applied Sciences and Arts  
Northwestern Switzerland, 4600 Olten, Switzerland  
kaspar.riesen@fhnw.ch

<sup>3</sup> Institute of Complex Systems, University of Applied Sciences and Arts Western  
Switzerland, 1700 Fribourg, Switzerland  
andreas.fischer@hefr.ch

**Abstract.** Biometric authentication by means of handwritten signatures is a challenging pattern recognition task, which aims to infer a writer model from only a handful of genuine signatures. In order to make it more difficult for a forger to attack the verification system, a promising strategy is to combine different writer models. In this work, we propose to complement a recent structural approach to offline signature verification based on graph edit distance with a statistical approach based on metric learning with deep neural networks. On the MCYT and GPDS benchmark datasets, we demonstrate that combining the structural and statistical models leads to significant improvements in performance, profiting from their complementary properties.

**Keywords:** Offline signature verification · Graph edit distance  
Metric learning · Deep convolutional neural network · Triplet network

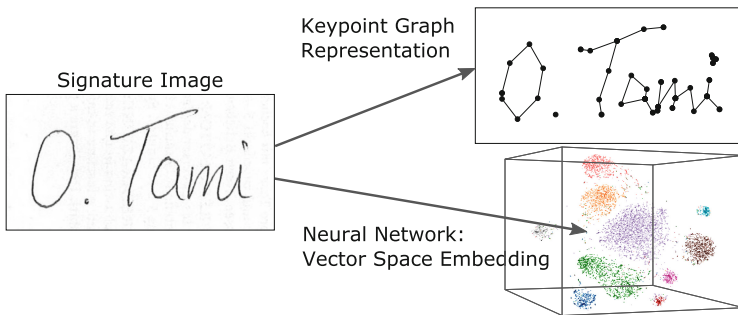
## 1 Introduction

To this day, handwritten signatures have remained a widely used and accepted means of biometric authentication. Automatic signature verification is an active field of research, accordingly, and the current state of the art achieves levels of accuracy similar to that of other biometric verification systems [12, 15]. Usually, two cases of signature verification are differentiated: the *offline* case, where only a static image of the signature is available, and the *online* case, where additional dynamic information like the velocity is available. Due to the lack of this information, offline signature verification applies to more use cases, but it is also considered the more challenging task.

Most state-of-the-art approaches to offline signature verification rely on statistical pattern recognition, i.e. signatures are represented using fixed-size feature vectors. These vector representations are often generated using handcrafted feature extractors leveraging either *local information*, such as local binary patterns, histogram of oriented gradients, or Gaussian grid features taken from signature contours [23], or *global information*, e.g. geometrical features like Fourier descriptors, number of branches in the skeleton, number of holes, moments, projections, distributions, position of barycenter, tortuosities, directions, curvatures and chain codes [15, 19]. More recently, with the advent of deep learning, we observe a shift away from handcrafted features towards learning features directly from the images using deep convolutional neural networks (CNN) [11].

Another way of approaching signature verification is by using graphs and structural pattern recognition. Graphs offer a more powerful representation formalism that can be beneficial for signature verification. For example, by capturing local information in nodes and their relations in the global structure using edges. But the representational power of graphs comes at the price of high computational complexity. This is probably why graphs have only been used rather rarely for signature verification in the past. Examples include the work of Sabourin et al. [22] (signatures represented based on stroke primitives), Bansal et al. [4] (modular graph matching approach), and Fotak et al. [9] (basic concepts of graph theory). More recently, a structural approach for signature verification has been introduced by Maergner et al. [16]. They propose a general signature verification framework based on the graph edit distance between labeled graphs. They employ a bipartite approximation framework [20] to reduce the computational complexity and report promising verification results using so-called keypoint graphs.

In this paper, we argue that structural and statistical signature models are quite different, with complementary strengths, and thus well-suited for multiple classifier systems. As illustrated in Fig. 1, we propose to combine the graph-based approach of Maergner et al. [16] with a statistical model inspired by recent advances in the field of deep learning, namely metric learning by means of a deep CNN [13] with the triplet loss function [14]. Such deep triplet networks can be



**Fig. 1.** Proposed structural and statistical signature image representations.

used to embed signature images in a vector space, where signatures of the same user have a small distance and signatures of different users have a large distance. To our knowledge, this is the first combination of a graph-based approach and a deep neural network based approach for the task of signature verification.

In the remainder, the structural approach is described in Sect. 2, the statistical approach in Sect. 3, and the proposed combined system in Sect. 4. Afterwards, we present our experimental results in Sect. 5 and draw conclusions in Sect. 6.

## 2 Structural Graph-Based Approach

The structural approach used in this paper has been proposed by Maergner et al. in [16]. Two signature images are compared by first binarizing and skeletonizing the image, then creating keypoint graphs from each skeleton image, and lastly comparing the two graphs using an approximation of the graph edit distance. In the following subsections, we briefly review these steps. For a more detailed description, see [16].

### 2.1 Keypoint Graphs

Formally, a labeled graph is defined as a four-tuple  $g = (V, E, \mu, \nu)$ , where  $V$  is the finite set of nodes,  $E \subseteq V \times V$  is the set of edges,  $\mu : V \rightarrow L_V$  is the node labeling function, and  $\nu : E \rightarrow L_E$  is the edge labeling function.

Keypoint graphs are created from points extracted from the skeleton image. Specifically, the nodes in the graph stand for certain points on the skeleton and are labeled with their coordinates. These points are end- and junction-points of the skeleton as well as additional points sampled in equidistant intervals of  $D$ . Unlabeled and undirected edges connect the nodes that are connected on the skeleton. The node labels are centered so that their average is  $(0, 0)$ . See Fig. 1 for an example of a keypoint graph.

### 2.2 Graph Edit Distance

Graph edit distance (GED) offers a way to compare any kind of labeled graph given an appropriate cost function. This makes GED one of the most flexible graph matching approaches. It calculates the cost of the lowest-cost edit path that transforms graph  $g_1 = (V_1, E_1, \mu_1, \nu_1)$  into graph  $g_2 = (V_2, E_2, \mu_2, \nu_2)$ . An edit path is a sequence of edit operations, for each of which a certain cost is defined. Commonly, substitutions, deletions, and insertions of nodes and edges are considered as edit operations. The main disadvantage of GED is its computational complexity since it is exponential in the number of nodes in the two graphs,  $O(|V_1|^{|V_2|})$ .

This issue can be addressed by using an approximation of GED. In this paper, the bipartite approximation framework proposed by Riesen and Bunke [20] is applied. The computation of GED is reduced to an instance of a *linear sum*



*assignment problem* with cubic complexity,  $O((V_1 + V_2)^3)$ . For signature verification, the lower bound introduced in [21] is considered.

The cost function is defined in the following way. The cost of a node substitution is the Euclidean distance between the node labels. For node deletion and insertion, a constant cost  $C_{\text{node}}$  is used. For edges, the substitution cost is set to zero. The edge deletion and insertion cost is set to a constant value  $C_{\text{edge}}$ .

Finally, the graph edit distance is normalized by dividing by the maximum graph edit distance, viz. the cost of deleting all nodes and edges from the first graph and inserting all the nodes and edges of the second graph. Thus, the graph-based dissimilarity is in  $[0, 1]$  and describes how large the graph edit distance is when compared with the maximum graph edit distance. Formally, the graph-based dissimilarity of two signature images is defined as follows:

$$d_{\text{GED}}(r, t) = \frac{\text{GED}(g_r, g_t)}{\text{GED}_{\text{max}}(g_r, g_t)}, \quad (1)$$

where  $g_r$  and  $g_t$  are the keypoint graphs of the signatures images  $r$  and  $t$  respectively,  $\text{GED}(g_r, g_t)$  is the lower bound of the graph edit distance between  $g_r$  and  $g_t$ , and  $\text{GED}_{\text{max}}(g_r, g_t)$  is the maximum graph edit distance between  $g_r$  and  $g_t$ .

### 3 Statistical Neural Network-Based Approach

We train a deep CNN [13] using a triplet-based learning method to embed images of signatures into a high-dimensional space where the distance of two signatures reflect their similarity, i.e. two signatures of the same user are close together and signatures from different users are far apart. An exemplary visualization of the vectors produced by such model is shown in Fig. 1, where points of the same class are grouped together in clusters. This approach has been investigated in the recent past for several image matching problems with promising success, including [3, 14, 24].

#### 3.1 Triplet-Based Learning

A triplet is a tuple of three signatures  $\{a, p, n\}$  where  $a$  is the anchor (reference signature),  $p$  is the positive sample (a signature from the same user) and  $n$  is the negative sample (a signature from another user). The neural network is then trained to minimize the loss function defined as:

$$L(\delta_+, \delta_-) = \max(\delta_+ - \delta_- + \mu, 0), \quad (2)$$

where  $\delta_+$  and  $\delta_-$  are the Euclidean distance between anchor-positive and anchor-negative pairs in the feature space and  $\mu$  is the margin used.

### 3.2 Signature Image Matching

We define the neural network as the function  $f$  that embeds a signature image into a latent space as previously described. The dissimilarity of two signature images  $r$  and  $t$  can now be defined as the Euclidean distance of their embedding vectors. Formally,

$$d_{\text{neural}}(r, t) = \|f(r) - f(t)\|_2. \quad (3)$$

## 4 Combined Signature Verification System

A signature verification system has to decide whether an unseen signature image is a genuine signature of the claimed user. This decision is being made by calculating a dissimilarity score between the *reference* signature of the claimed user and the unseen signature. The signature is accepted if this dissimilarity score (see Eqs. 5 or 6) is below a certain threshold, otherwise the signature is rejected.

### 4.1 User-Based Normalization

It is expected that the users have different intra-user variability. Therefore, each dissimilarity score is normalized using the average dissimilarity score between the reference signatures of the current user as suggested in [16]. Formally,

$$\hat{d}(r, t) = \frac{d(r, t)}{\delta(R)}, \quad (4)$$

where  $t$  is a questioned signature image,  $r \in R$  is a reference signature image,  $R$  is the set of all reference signature images of the current users, and

$$\delta(R) = \frac{1}{|R|} \sum_{r \in R} \min_{s \in R \setminus r} d(r, s).$$

### 4.2 Signature Verification Score

The minimum dissimilarity over all reference signatures  $R$  of the claimed user to the questioned signature  $t$  is used as signature verification score. Formally,

$$d(R, t) = \min_{r \in R} \hat{d}(r, t) \quad (5)$$

### 4.3 Multiple Classifier System

We propose a multiple classifier system (MCS) as a linear combination of the graph-based dissimilarity and the neural network based dissimilarity. Z-score normalization based on all reference signature images in the current data set is applied to each dissimilarity score before the combination. Formally, we define

$$d_{\text{MCS}}(R, t) = \min_{r \in R} \left( \hat{d}_{\text{GED}}^*(r, t) + \hat{d}_{\text{neural}}^*(r, t) \right), \quad (6)$$

where  $\hat{d}^*$  is the z-score normalized dissimilarity score.

## 5 Experimental Evaluation

We evaluate the performance on two publicly available benchmark data sets by measuring the *equal error rate* (EER). The EER is the point where the false acceptance rate and the false rejection rate are equal in the *detection error tradeoff* (DET) curve. Two kinds of forgeries are tested: *skilled forgeries* (SF), which are forgeries created with information about the user’s signature, and so-called *random forgeries*<sup>1</sup> (RF), which are genuine signatures of other users that are used in a brute force attack.

### 5.1 Data Sets

In our evaluation, we use the following publicly available signature data sets:

- **GPDSsynthetic-Offline:** Ferrer et al. introduced this data set in [5]. It contains 24 genuine signatures and 30 skilled forgeries for 4,000 synthetic users. This data set replaces previous signatures databases from the GPDS group, which are not available anymore. We use four subsets of this data set: one containing the first 75 users, and three containing the last 10, 100, or 1000 users. These subsets are called *GPDS-75*, *GPDS-last10*, *GPDS-last100*, and *GPDS-last1000* respectively.
- **MCYT-75:** This data set is part of the MCYT baseline corpus introduced by Ortega-Garcia et al. in [7, 18]. It contains 75 users with 15 genuine signatures and 15 skilled forgeries each.

### 5.2 Tasks

We distinguish two tasks depending on the number of references available for each user. Five genuine signatures per user (*R5*) or ten genuine signatures per user (*R10*). In both cases, the remaining genuine signatures are used for testing in both the skilled forgery (SF) and in the random forgery (RF) evaluation. The SF evaluation is performed using all available skilled forgeries for each user. The RF evaluation is carried out using the first genuine signature of all other users in the data set as random forgeries. For example for the GPDS-75 R10 tasks, that gives us  $75 \cdot 10 = 750$  reference signatures,  $75 \times 14 = 1,050$  genuine signatures,  $75 \times 30 = 2,250$  skilled forgeries, and  $75 \times 74 = 5,550$  random forgeries.

### 5.3 Setup

**Graph Parameter Validation.** For the keypoint graph extraction, we use  $D = 25$ , which has been proposed in [16]. The cost function parameters  $C_{\text{node}}$  and  $C_{\text{edge}}$  are validated on the GPDS-last100 data set using the random forgery evaluation. No skilled forgeries are used. We perform a grid search over  $C_{\text{node}} \in \{10, 15, \dots, 60\}$  and  $C_{\text{edge}} \in \{10, 15, \dots, 60\}$ . The best results have been achieved using  $C_{\text{node}} = 25$  and  $C_{\text{edge}} = 45$ . We use these parameters in our experiments on GPDS-75 and MCYT-75.

<sup>1</sup> This term is mainly used in the pattern recognition community and it might be confusing for readers from other fields. For more details, see [17].

**Neural Network Training.** We use the ResNet18 architecture [13], which is an 18 layer deep variant of a convolutional neural network that uses shortcut connections between layers to tackle the vanishing gradient problem.

We train three different models using the DeepDIVA<sup>2</sup> framework [1] for the task of embedding the signature images in the vector space, where each of the models differs with respect to how much data is used for training (GPDS-last10, GPDS-last100, or GPDS-last1000). We call these systems NN-last10, NN-last100, and NN-last1000 respectively. For each person in the data set, there are 24 genuine images. We use 16 of them for training and the remaining 8 for validating the performance of the model. Skilled forgeries are not used for training.

The network is trained using the Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.01 and momentum of 0.9.

### 5.4 Results on MCYT-75 and GPDS-75

The EER results on GPDS-75 and MCYT-75 for both RF and SF are shown in Table 1. In all but one case, the combination of the GED approach and the neural network achieves better results than the best individual system. The neural networks trained on GPDS-last100 and GPDS-last1000 are on its own significantly better on the RF task. We can see that NN-last1000 is more specialized on the RF task on the GPDS-75 data set while losing performance on the MCYT-75 data set. Two DET curves are shown in Fig. 2.

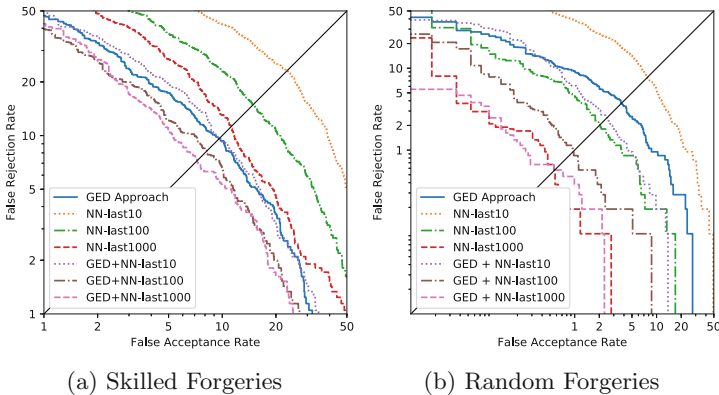


Fig. 2. DET curves for GPDS-75 R10.

### 5.5 Comparison with State-of-the-Art

Many different evaluation protocols are used for signature verification. To allow a fair comparison, we have to follow the same protocol. In the following, we present EER results using two different protocols and compare our results with other published results.

<sup>2</sup> <https://github.com/DIVA-DIA/DeepDIVA> (April 29, 2018).

**Table 1.** EER on GPDS-75/MCYT-75. Results on skilled forgeries (SF) and on random forgeries (RF) using the first 5 or 10 genuine as references (R5/R10).

System	GPDS-75				MCYT-75			
	RF		SF		RF		SF	
	R5	R10	R5	R10	R5	R10	R5	R10
GED approach	4.90	3.71	11.69	9.60	5.86	2.65	20.09	13.60
NN-last10	10.40	7.71	25.87	23.11	6.47	4.79	19.56	17.16
GED + NN-last10	4.00	2.47	12.04	9.51	3.19	1.59	16.53	11.29
NN-last100	3.28	2.05	17.96	14.84	3.59	1.59	20.36	12.80
GED + NN-last100	2.16	0.95	9.82	8.18	<b>2.79</b>	1.41	<b>15.56</b>	<b>10.40</b>
NN-last1000	0.68	<b>0.56</b>	13.29	11.20	3.73	1.15	19.02	13.78
GED + NN-last1000	<b>0.65</b>	<b>0.56</b>	<b>9.24</b>	<b>7.24</b>	2.92	<b>0.79</b>	17.69	11.11

**Table 2.** Comparison on GPDS-75/MCYT-75. Average EER results over 10 random selections of ten reference signatures. Evaluated on GPDS-75 and MCYT-75 for random forgeries (RF) and skilled forgeries (SF).

System	GPDS-75 R10		MCYT-75 R10	
	RF	SF	RF	SF
Ferrer et al. [6] (see footnote 4)	0.76*	16.01	<b>0.35*</b>	11.54
Maergner et al. [16]	2.73	8.29	2.83	12.01
Proposed GED approach	2.75	8.31	2.67	11.42
Proposed NN-last1000	0.44	10.79	1.57	12.24
Proposed GED + NN-last1000	<b>0.41</b>	<b>6.49</b>	1.05	<b>9.15</b>

\*: All genuine signatures of other users as RF.

**Table 3.** Comparison on MCYT-75 R5/R10. EER results for skilled forgeries (SF) and random forgeries (RF) using an *a posteriori* user-dependent score normalization. The first 5 or 10 genuine signatures are used as references for R5 and R10 respectively.

System	MCYT-75 R5		MCYT-75 R10	
	RF	SF	RF	SF
Alonso-Fernandez et al. [2]	9.79*	23.78	7.26*	22.13
Fierrez-Aguilar et al. [7]	2.69**	11.00	1.14**	9.28
Gilperez et al. [10]	2.18*	<b>10.18</b>	1.18*	<b>6.44</b>
Maergner et al. [16]	2.40	14.49	1.89	11.64
Proposed GED approach	2.45	14.84	1.89	12.27
Proposed NN-last100	2.14	15.02	1.77	13.16
Proposed GED + NN-last100	<b>0.92</b>	10.67	<b>0.25</b>	10.13

\*: All genuine signatures of other users as RF.

\*\* : First 5 genuine signatures from each other user as RF.

**Comparison on GPDS-75 and MCYT-75.** This evaluation is performed by selecting 10 reference signatures randomly<sup>3</sup> and average the results over 10 runs. Table 2 shows our results using the same protocol compared with the previously published results: results published in [16] and results presented on the GPDS website<sup>4</sup>, which have been achieved using the system published in [6]. The proposed combination of the GED approach and NN-last1000 achieves the lowest EER in all tasks except for random forgeries on MCYT-75.

**Comparison on MCYT-75.** A group of publications has presented results on the MCYT-75 data set using the a posteriori user-depended score normalization introduced in [8]. By applying this normalization, all user scores are aligned so that the EER threshold is the same for all users. Table 3 shows the published results as well as our results using the same normalization. The combination of GED and NN-last100 achieves results in the middle ranks for the SF task and the overall best results for the RF task.

## 6 Conclusions and Outlook

Combining structural and statistical models has significantly improved the signature verification performance on the MCYT-75 and GPDSsynthetic-Offline benchmark datasets. The structural model based on approximate graph edit distance achieved better results against skilled forgeries, while the statistical model based on metric learning with deep triplet networks achieved better results against a brute-force attack with random forgeries. The proposed system was able to combine these complementary strengths and has proven to generalize well to unseen users, which have not been used for model training and hyperparameter optimization.

We can see several lines of future research. For the structural method, more graph-based representations and cost functions may be explored in the context of graph edit distance. For the statistical method, synthetic data augmentation may lead to a more accurate vector space embedding. Finally, we believe that there is a great potential in combining even more structural and statistical classifiers into one large multiple classifier system. Such a system is expected to further improve the robustness of biometric authentication.

**Acknowledgment.** This work has been supported by the Swiss National Science Foundation project 200021\_162852.

---

<sup>3</sup> We use the same random selections for all our results.

<sup>4</sup> <http://www.gpds.ulpgc.es/downloadnew/download.htm> (April 29, 2018).

## References

1. Alberti, M., Pondenkandath, V., Würsch, M., Ingold, R., Liwicki, M.: DeepDIVA: a highly-functional python framework for reproducible experiments. In: International Conference on Frontiers in Handwriting Recognition (2018, submitted)
2. Alonso-Fernandez, F., Fairhurst, M., Fierrez, J., Ortega-Garcia, J.: Automatic measures for predicting performance in off-line signature. In: Proceedings of the 14th International Conference on Image Processing, pp. 369–372 (2007)
3. Balntas, V., Riba, E., Ponsa, D., Mikolajczyk, K.: Learning local feature descriptors with triplets and shallow convolutional neural networks. In: Proceedings of the British Machine Vision Conference (BMVC), September 2016
4. Bansal, A., Gupta, B., Khandelwal, G., Chakraverty, S.: Offline signature verification using critical region matching. *Int. J. Sig. Process. Image Process. Pattern Recogn.* **2**(1), 57–70 (2009)
5. Ferrer, M.A., Diaz-Cabrera, M., Morales, A.: Static signature synthesis: a neuro-motor inspired approach for biometrics. *IEEE Trans. Pattern Anal. Mach. Intell.* **37**(3), 667–680 (2015)
6. Ferrer, M.A., Vargas, J.F., Morales, A., Ordonez, A.: Robustness of offline signature verification based on gray level features. *IEEE Trans. Inf. Forensics Secur.* **7**(3), 966–977 (2012)
7. Fierrez-Aguilar, J., Alonso-Hermira, N., Moreno-Marquez, G., Ortega-Garcia, J.: An off-line signature verification system based on fusion of local and global information. In: Maltoni, D., Jain, A.K. (eds.) *BioAW 2004*. LNCS, vol. 3087, pp. 295–306. Springer, Heidelberg (2004). [https://doi.org/10.1007/978-3-540-25976-3\\_27](https://doi.org/10.1007/978-3-540-25976-3_27)
8. Fierrez-Aguilar, J., Ortega-Garcia, J., Gonzalez-Rodriguez, J.: Target dependent score normalization techniques and their application to signature verification. *IEEE Trans. Syst. Man. Cybern. Part C* **35**(3), 418–425 (2004)
9. Fotak, T., Baca, M., Koruga, P.: Handwritten signature identification using basic concepts of graph theory. *WSEAS Trans. Sig. Process.* **7**(4), 145–157 (2011)
10. Gilperez, A., Alonso-Fernandez, F., Pecharroman, S., Fierrez, J., Ortega-Garcia, J.: Off-line signature verification using contour features. In: Proceedings of the 11th International Conference on Frontiers in Handwriting Recognition, pp. 1–6 (2008)
11. Hafemann, L.G., Sabourin, R., Oliveira, L.S.: Learning features for offline handwritten signature verification using deep convolutional neural networks. *Pattern Recogn.* **70**, 163–176 (2017)
12. Hafemann, L.G., Sabourin, R., Oliveira, L.S.: Offline handwritten signature verification - literature review. In: Proceedings of International Conference on Image Processing Theory, Tools and Applications (IPTA), pp. 1–8 (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
14. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: Feragen, A., Pelillo, M., Loog, M. (eds.) *SIMBAD 2015*. LNCS, vol. 9370, pp. 84–92. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-24261-3\\_7](https://doi.org/10.1007/978-3-319-24261-3_7)
15. Impedovo, D., Pirlo, G.: Automatic signature verification: the state of the art. *IEEE Trans. Syst. Man Cybern. Part C* **38**(5), 609–635 (2008)
16. Maergner, P., Riesen, K., Ingold, R., Fischer, A.: A structural approach to offline signature verification using graph edit distance. In: Proceedings of International Conference on Document Analysis and Recognition, pp. 1216–1222. IEEE (2017)

17. Malik, M.I., Liwicki, M.: From terminology to evaluation: performance assessment of automatic signature verification systems. In: Proceedings of International Conference on Frontiers in Handwriting Recognition, pp. 613–618 (2012)
18. Ortega-Garcia, J., et al.: MCYT baseline corpus: a bimodal biometric database. *IEEE Proc.-Vis. Image Sig. Process.* **150**(6), 395–401 (2003)
19. Plamondon, R., Lorette, G.: Automatic signature verification and writer identification - the state of the art. *Pattern Recogn.* **22**(2), 107–131 (1989)
20. Riesen, K., Bunke, H.: Approximate graph edit distance computation by means of bipartite graph matching. *Image Vis. Comput.* **27**(7), 950–959 (2009)
21. Riesen, K., Fischer, A., Bunke, H.: Computing upper and lower bounds of graph edit distance in cubic time. In: El Gayar, N., Schwenker, F., Suen, C. (eds.) *ANNPR 2014. LNCS (LNAI)*, vol. 8774, pp. 129–140. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-11656-3\\_12](https://doi.org/10.1007/978-3-319-11656-3_12)
22. Sabourin, R., Plamondon, R., Beaumier, L.: Structural interpretation of handwritten signature images. *Int. J. Pattern Recog. Artif. Intell.* **8**(3), 709–748 (1994)
23. Yilmaz, M.B., Yanikoglu, B., Tirkaz, C., Kholmatov, A.: Offline signature verification using classifier combination of HOG and LBP features. In: Proceedings of the International Joint Conference on Biometrics, pp. 1–7 (2011)
24. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 4353–4361 (2015)





# On Association Graph Techniques for Hypergraph Matching

Giulia Sandi<sup>1</sup>(✉), Sebastiano Vascon<sup>1,2</sup>, and Marcello Pelillo<sup>1,2</sup>

<sup>1</sup> Department of Environmental Sciences, Informatics and Statistics,  
Ca' Foscari University of Venice, Venice, Italy  
giuliasandi@hotmail.com, {sebastiano.vascon,pelillo}@unive.it

<sup>2</sup> European Centre for Living Technology, Venice, Italy

**Abstract.** Association graph techniques represent a classical approach to tackle the graph matching problem and recently the idea has been generalized to the case of hypergraphs. In this paper, we explore the potential of this approach in conjunction with a class of dynamical systems derived from the Baum-Eagon inequality. In particular, we focus on the pure isomorphism case and show, with extensive experiments on a large synthetic dataset, that despite its simplicity the Baum-Eagon dynamics does an excellent job at finding globally optimal solutions.

**Keywords:** Hypergraph isomorphism · Association graph  
Baum-Eagon inequality · Polynomial optimization

## 1 Introduction

The problem of hypergraph (as opposed to graph) matching has gained increasing attention in the last few years, thanks to the advantages that arise from considering relationships among more than two elements, thus encoding an higher pool of information. Dealing with these topics is of particular interest in fields such as computer vision, pattern recognition and machine learning, due to the need of solving problems such as, e.g., object recognition, feature tracking, shape matching and scene registration, where high-relations are naturally used. Different studies have transformed this problem into an optimization one: maximizing the sum of the matching scores (see e.g. [5, 9, 16] and the references therein).

The isomorphism problem on graphs has been successfully addressed in [12, 13] using the classical approach of computing the *association graph* from the two structures being matched and then applying techniques from evolutionary game theory on the newly built graph. Recently a similar approach has been applied on hypergraphs in [7, 17], using dynamics inspired from the Baum-Eagon inequality [2, 11, 15]. The authors of the aforementioned papers obtained good results on uniform hypergraphs of cardinality 3 (aka 3-graphs), but the developed approaches can easily be applied to structures of larger cardinality.

Motivated by these recent works, in this paper we aim to systematically explore the potential of this approach on the simplest version of the hypergraph matching problem, namely the isomorphism case. In particular, we have performed a series of experiments on a synthetic dataset made of 900 uniform 3-graphs of different orders, randomly generated with various connectivities. The results obtained are impressive: the proposed framework correctly identifies 100% of the isomorphisms, with all the different dynamics tested.

The outline of the article is as follows. Section 2 presents the definition of association hypergraph and some fundamental results needed to use this auxiliary structure in order to solve the isomorphism problem. In Sect. 3 we introduce the Baum-Eagon inequality with the related dynamics, also in their exponential form. In Sect. 4, experimental results are presented, which turned out to be impressive in terms of precision. Finally, Sect. 5 concludes the article.

## 2 Hypergraph Matching Using Association Hypergraphs

A hypergraph is formally defined as a pair  $H = (V, E)$ , where  $V$  is the (finite) set of vertices and  $E \subseteq 2^V$  is the set of hyperedges (where  $2^V$  denotes the powerset of  $V$ ). Even though hypergraphs may have hyperedges of different cardinalities, we will focus in this paper only on *uniform hypergraphs*, or  $k$ -graphs, whose edges have fixed cardinality  $k \geq 2$ . Trivially the case  $k = 2$  represents classical graphs, in which only pairwise relations are taken into account. The order of  $H$  is the number of its vertices, while its size is the number of hyperedges. Given  $k$  vertices  $i_1, \dots, i_k \in V$ , they are said to be adjacent if  $\{i_1, \dots, i_k\} \in E$ . The *degree* of a vertex  $i \in V$ , denoted by  $\deg(i)$ , is the number of vertices adjacent to it. From now on we will indifferently use either the word graph or hypergraph, referring always to uniform hypergraphs, except where confusion may arise.

Given two hypergraphs  $H' = (V', E')$  and  $H'' = (V'', E'')$ , an isomorphism between them is defined by any bijection  $\phi : V' \rightarrow V''$  for which  $\{i_1, \dots, i_k\} \in E' \Leftrightarrow \{\phi(i_1), \dots, \phi(i_k)\} \in E'', \forall i_1, \dots, i_k \in V'$ . If an isomorphism exists between two hypergraphs, then they are said to be isomorphic. Therefore, solving the graph isomorphism problem means deciding whether at least one isomorphism exists between two graphs, and in this case to find one. The hypergraph matching problem is more general and difficult [6], and includes the graph isomorphism problem as a special case. It consists of finding a match between the largest subset of vertices of  $H'$  and  $H''$ , such that the subgraphs defined by these subsets of nodes are isomorphic. Finding a maximal common subgraph, that is an isomorphism between subgraphs that is not included in any larger subgraph isomorphism, is a simple version of the hypergraph matching problem.

The notion of *association graph*, a useful auxiliary graph structure designed for solving general graph matching problems, has been introduced in [1] and also in [8], and can be easily generalised to uniform hypergraphs.

**Definition 1.** The association hypergraph derived from unweighted uniform hypergraphs  $H' = (V', E')$  and  $H'' = (V'', E'')$  is the undirected unweighted

hypergraph  $H = (V, E)$  defined as:

$$V = V' \times V''$$

and

$$E = \{ \{ (i_1, j_1), \dots, (i_k, j_k) \} \in V' \times V'' : \\ i_1 \neq \dots \neq i_k, j_1 \neq \dots \neq j_k, \\ \{ i_1, \dots, i_k \} \in E' \Leftrightarrow \{ j_1, \dots, j_k \} \in E'' \}.$$

Given a  $k$ -graph  $H = (V, E)$ , a *clique* is defined as a subset of vertices  $C$  such that for all distinct  $i_1, \dots, i_k \in C$ , they are mutually adjacent, that is  $\{i_1, \dots, i_k\} \in E$ . A *maximal* clique is defined as a clique that is not contained in any larger clique, while a *maximum* clique is defined as the largest clique in the graph. The cardinality of the maximum clique is called the clique number  $\omega(H)$ .

In the following result, which generalises to hypergraphs an analogous result obtained in [12, 13] for graphs, a one-to-one correspondence between the graph isomorphism problem and the maximum clique problem is demonstrated.

**Theorem 1.** *Let  $H' = (V', E')$  and  $H'' = (V'', E'')$  be two hypergraphs of order  $n$  and edge cardinality  $k$ , and let  $H = (V, E)$  be the related association  $k$ -graph. Then  $H'$  and  $H''$  are isomorphic if and only if  $\omega(H) = n$ . In this case, any maximum clique of  $H$  induces an isomorphism between  $H'$  and  $H''$ , and vice versa. In general, maximum and maximal common subgraph isomorphisms between  $H'$  and  $H''$  are in one-to-one correspondence with maximum and maximal cliques in  $H$ , respectively.*

*Sketch of proof.* Suppose that the two  $k$ -graphs are isomorphic, and let  $\phi$  be an isomorphism between them. Then the subset of vertices of  $H$  defined as  $C_\phi = \{ (i, \phi(i)) : \forall i \in V' \}$  is clearly a maximum clique of cardinality  $n$ . In reverse, let  $C$  be an  $n$ -vertex maximum clique of  $H$ , and for each  $(i, h) \in C$  define  $\phi(i) = h$ . Then it is easy to see that  $\phi$  is an isomorphism between  $H'$  and  $H''$  because of the way the association  $k$ -graph is constructed. The proof for the general case is analogous.

Consider an arbitrary undirected hypergraph of order  $n$ ,  $H = (V, E)$ , and let  $S_n$  denotes the standard simplex of  $\mathbb{R}^n$ :

$$S_n = \left\{ \mathbf{x} \in \mathbb{R}^n : x_i \geq 0, \text{ for all } i = 1, \dots, n, \text{ and } \sum_{i=1}^n x_i = 1 \right\} \quad (1)$$

Given a subset of vertices  $C$  of  $H$ , its characteristic vector is denoted by  $\mathbf{x}^c$ , and it represents the point in  $S_n$  defined as:

$$x_i^c = \begin{cases} 1/|C|, & \text{if } i \in C \\ 0, & \text{otherwise} \end{cases}$$

where  $|C|$  indicates the cardinality of  $C$ .

Now, consider the *Lagrangian* of  $H$ , which is the polynomial function defined as:

$$f(\mathbf{x}) = \sum_{e \in E} \prod_{i \in e} x_i \tag{2}$$

A point  $\mathbf{x}^* \in S_n$  is said to be a global maximizer of  $f$  in  $S_n$  if  $f(\mathbf{x}^*) \geq f(\mathbf{x})$ , for all  $\mathbf{x} \in S_n$ . On the other hand it is said to be a local maximizer if there exists an  $\epsilon > 0$  such that  $f(\mathbf{x}^*) \geq f(\mathbf{x})$  for all  $\mathbf{x} \in S_n$  whose distance from  $\mathbf{x}^*$  is less than  $\epsilon$ . If  $f(\mathbf{x}^*) = f(\mathbf{x})$  implies  $\mathbf{x}^* = \mathbf{x}$ , then  $\mathbf{x}^*$  is said to be a strict local maximizer.

For the case of a graph  $G$  (namely an hypergraph with  $k = 2$ ), the Motzkin-Straus theorem [10] establishes a remarkable connection between global (local) maximizers of the Lagrangian in  $S_n$  and maximum (maximal) cliques of  $G$  itself. In particular, it asserts that in  $G$ , a subset  $C$  of its vertices is a maximum clique if and only if a global maximizer of the Lagrangian of the graph  $G$  in the standard simplex  $S_n$  is the characteristic vector  $\mathbf{x}^C$ .

The formulation of  $f(\mathbf{x})$  given in Eq. (2) is the same used in [7, 17] and is motivated by the Motzkin-Straus theorem on graphs. Therefore we are going to focus on this function in our experiments. However different formulations are possible, for example the one proposed in [14].

### 3 Finding Isomorphisms Using the Baum-Eagon Dynamics

Given the definitions and results in the previous section, we can reduce the problem of finding an isomorphism between two hypergraphs to the following (linearly constrained) polynomial optimization problem:

$$\begin{aligned} &\text{maximize} && f(\mathbf{x}) = \sum_{e \in E} \prod_{i \in e} x_i \\ &\text{subject to} && \mathbf{x} \in S_n \end{aligned} \tag{3}$$

A simple and effective way of optimizing this function is to use a result introduced by Baum and Eagon [2] in the late 1960s. They presented a class of non-linear transformations in the standard simplex, and proved a central result, generalizing a previous one introduced by Blakley [4] on related characteristics for particular homogeneous quadratic transformations. The following theorem presents a result known as the Baum-Eagon inequality.

**Theorem 2.** (Baum-Eagon [2]) *Let  $Q(\mathbf{x})$  be a homogeneous polynomial in the variables  $x_j$  with non-negative coefficients, and let  $\mathbf{x} \in \Delta$ . Define the mapping  $\mathbf{z} = \mathcal{M}(\mathbf{x})$  from  $\Delta$  to itself as follow:*

$$z_j = x_j \frac{\partial Q(\mathbf{x})}{\partial x_j} \bigg/ \sum_{l=1}^n x_l \frac{\partial Q(\mathbf{x})}{\partial x_l}, \quad j = 1, \dots, n. \tag{4}$$

*Then  $Q(\mathcal{M}(\mathbf{x})) > Q(\mathbf{x})$  unless  $\mathcal{M}(\mathbf{x}) = \mathbf{x}$ .*

A continuous mapping as the one defined in this theorem is known as a *growth transformation*. Interestingly, only first-order derivatives are used in the definition of the mapping  $\mathcal{M}$ , that is yet able to increase  $Q$  taking a finite number of steps, being in this way sharply in contrast with classical gradient methods that need to compute high-order derivatives in order to define the size of the infinitesimal steps to be taken. Moreover gradient descend need to perform some projection operator, causing some problems for points on the boundary. Instead, with Theorem 2, only a computationally easy normalization on rows is needed.

For these reasons we can affirm that the Baum-Eagon inequality supplies a powerful tool for maximizing polynomials functions in the standard simplex, and in fact they have been used as a main component for different statistical estimation techniques developed within the theory of probabilistic functions of Markov Chains [3], as well as for analysing the dynamical properties of relaxation labelling processes [11]. Looking at the problem in Eq. (3), we can easily see that  $f$  is indeed an uniform polynomial with non-negative coefficients that have to be maximized in the standard simplex, so Theorem 2 can be applied to optimize it.

Paraphrasing the Baum-Eagon inequality we can formalize the following discrete time dynamic:

$$x_j(t + 1) = x_j(t) \frac{\delta_j(\mathbf{x})}{\sum_{i=1}^n x_i \delta_i(\mathbf{x})}, \quad j = 1 \dots n, \tag{5}$$

where for readability reasons we have defined  $\delta_j(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial x_j}$ .

Starting at time 0 with  $\mathbf{x}(0)$  inside the standard simplex  $S^n$ , the dynamic in Eq. 5 iteratively updates the state vector until convergence. At the end of the process the vector state will be in the form of a characteristic vector, thus thresholding it with respect to a small amount close to zero, will return only the elements of the association hypergraphs that belong to the (maximum) clique. As we will see in the results section, even though there is no theoretical guarantee that the discrete time dynamic just presented reaches the optimal maximizer of the function, the results of our experiments concerning isomorphism problems show that the basin of attraction of the global maximum are quite large, so that the dynamic in Eq. 5 always returned the maximum clique in the associations graph, and never incurred in local solutions.

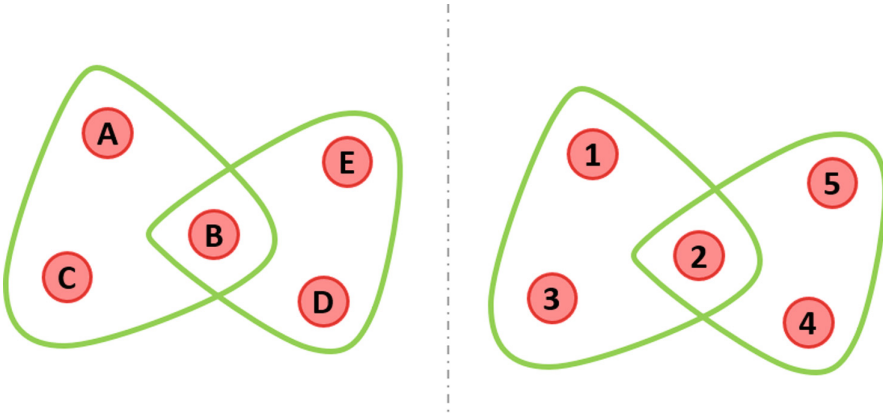
Moreover, in order to obtain a faster convergence, an exponential version of the dynamic can be defined as:

$$x_j(t + 1) = x_j(t) \frac{e^{\kappa \delta_j(\mathbf{x})}}{\sum_{i=1}^n x_i(t) e^{\kappa \delta_i(\mathbf{x})}}, \quad j = 1 \dots n. \tag{6}$$

Clearly, even though this exponential dynamic might decrease the time needed to find the clique, it introduces a new parameter  $\kappa$  that has to be tuned. This parameter has to be set in a way such that the optimization process is speeded up while guaranteeing the correctness of the results. In the following section some remarks are made on how the value of  $\kappa$  influences the search for the global maximum.

## 4 Experimental Results

The proposed approach is tested on random hypergraphs of different sizes, with different connectivities, in order to estimate its validity, and to understand if there are substantial differences in the results for hypergraphs that differ on these parameters. Hypergraphs of cardinality  $k = 3$  have been taken into consideration for computational reasons, however the framework can be applied also to  $k$ -graphs with  $k > 3$ .



**Fig. 1.** A pair of isomorphic 3-graphs.

The choice of using random graphs to test our framework has been made for a couple of different reasons. First, random graphs are not bound to any specific application, thus giving the possibility to test extensively all the variety of parameters combinations, even the ones that may be uncommon in some specific application, but still of interest. Second, they provide an experimental system that is easy to replicate and can therefore be used to make comparisons with other algorithms.

Experiments were made on randomly generated 3-graphs with 25 and 50 nodes and connectivities in the range [0.01%, 0.99%]. For each combination of these parameters, 30 graphs have been generated, and their vertices randomly permuted, in order to obtain a pair of isomorphic hypergraphs, performing a total of 900 different experiments. Each experiment has been tested with the standard Baum-Eagon dynamic (see Eq. (5)) and with its exponential version (see Eq. (6)) with the  $\kappa$  parameter ranging in  $\{10, 25, 50\}$ .

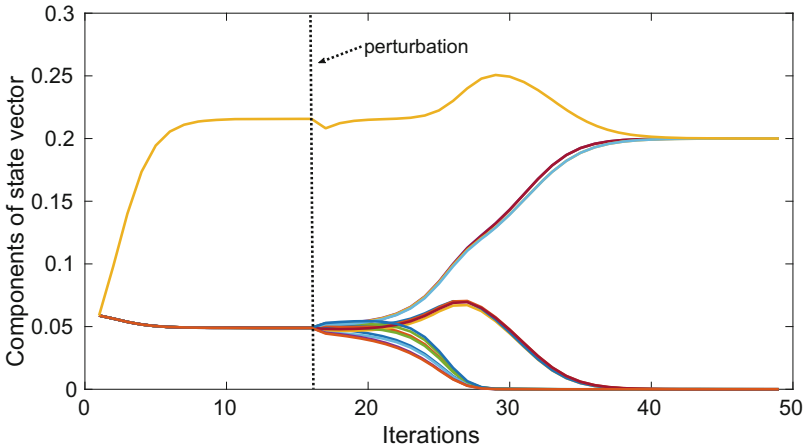
The algorithm was started in the barycentre of the simplex and stopped when either the distance of two subsequent points was smaller than a given threshold, set to  $10^{-10}$ , or when a maximum number of time-steps, equal to 1000, has been processed. When the algorithm stops, we check if a clique has been found: in the negative case, the final point is perturbed and the algorithm is started again,

in order to escape from saddle points. All the experiments have been run on a workstation equipped with an Intel Core i7-6800K at 3.40 GHz with 128 GB of RAM.

Since the size of the association graph increases exponentially with both the number of nodes in the hypergraphs to be matched and the cardinalities of the hyperedges involved, some pruning has been done on all the possible associations, so as to keep the order of the association hypergraph as small as possible. In particular the vertex set was constructed as follow:

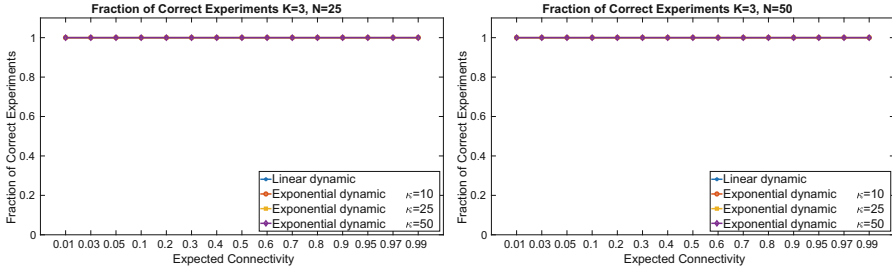
$$V = \{(i, j) \in V' \times V'' : deg(i) = deg(j)\}$$

and the edge set has been defined as in Definition 1. When the two graphs are isomorphic, Theorem 1 continues to hold, since the isomorphism preserves the degree property of vertices. However, this simple heuristic greatly decreases the order of the association graph, and therefore its size, notably easing the optimization task. In particular with  $n = 25$ , in the best case, that is when the connectivity rate is 0.5, only about the 7% of all the possible associations are created, while in the worst case, considering the extreme connectivity rates, only around the 20% of the associations are taken into consideration.



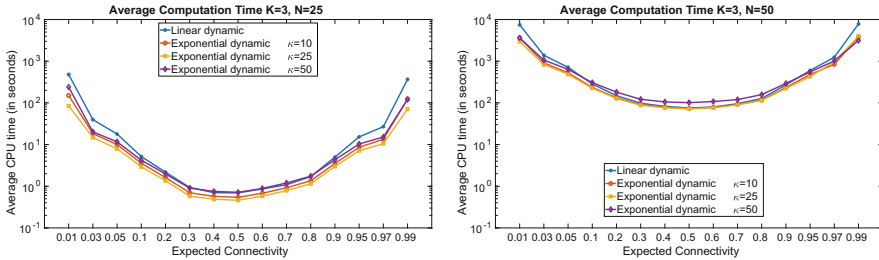
**Fig. 2.** Evolution through time of the components of the state vector  $\mathbf{x}(t)$  from the hypergraphs in Fig. 1 using the Baum-Eagon inequality. A perturbation can be seen at iteration 17 in order to escape a saddle point. After the perturbation the algorithm clearly makes a decision about which associations have to be chosen and which others have to be discarded.

Each pair of isomorphic graphs was given as input to the Baum-Eagon dynamic; after convergence, a success was recorded only when the cardinality of the returned clique was equal to the order of the graphs given as input. Because of the stopping criterion employed, this guarantees that a maximum clique, and therefore a correct isomorphism, was found.



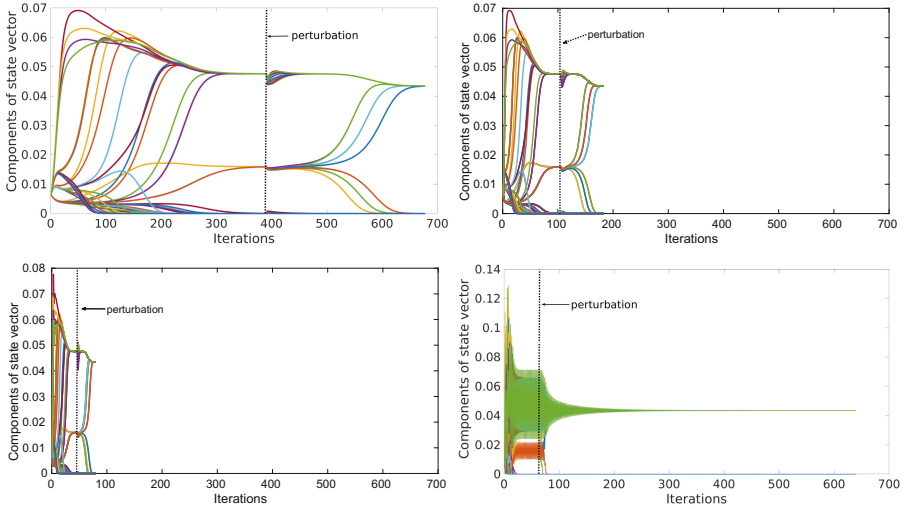
**Fig. 3.** All the experiments on hypergraphs with 25 (left) and 50 (right) nodes have correctly identified the isomorphism, with all the dynamics tested.

As we can see in Fig. 3, the obtained results are impressive in terms of correctness: all the isomorphisms have been properly found, with the algorithm returning 100% of the nodes in both graphs exactly coupled, for all the 900 experiments, independently of the dynamic that has been used. For what concerns time complexity, we can see in Fig. 4 that all the dynamics involved have the same behaviour, being extremely slow when dealing with very sparse or very dense graphs. With no surprise, we see that dealing with smaller hypergraphs results in shorter execution times, nevertheless the behaviour of the curves according to all the other parameters is exactly the same in both figures. However we can clearly see that in both cases the exponential dynamic with  $\kappa = 25$  is faster than all the other dynamics, outperforming the standard Baum-Eagon inequality of nearly one order of magnitude in the extreme connectivity rates, thus resulting to be really attractive, from a computational point of view. However, even though the exponential version of the dynamic might be faster, it involves setting the additional parameter  $\kappa$ . This operation is not trivial, since there is no theory about the correct way of choosing this parameter: the correct balance between



**Fig. 4.** Mean CPU time needed to run the optimization algorithm for finding isomorphism on hypergraphs with 25 (left) and 50 (right) nodes. Note that the y-axes are in logarithmic scale. The indicated timings include only the time needed to perform the optimization dynamics in order to find the clique. The time needed to compute the association hypergraph has not been taken into account since it is negligible with respect to the time needed for the optimization.





**Fig. 5.** Different behaviour of the dynamical systems under examination. On top left, the standard Baum-Eagon dynamic take about 700 iterations to converge; on top right, the exponential dynamic with  $\kappa = 10$  takes less than 200 iterations; on bottom left the exponential dynamic with  $\kappa = 25$  takes only 80 iterations to converge; on bottom right, due to the great oscillations the exponential dynamic with  $\kappa = 50$  takes again nearly 700 oscillations.

speed and stability has to be found, and even though the size of the association hypergraph has to be taken into consideration when choosing  $\kappa$ , it is not the only thing to mind. Figure 5 shows the evolution of the state vector through time using different values for the parameter  $\kappa$ . As we can see, with  $\kappa = 50$  the dynamic still converges to the maximum clique, but making many oscillations, thus needing a lot of iterations to return the correct result, explaining in this way why the exponential dynamic with this value of the parameter takes even longer that the standard Baum-Eagon dynamic in some cases.

## 5 Conclusions

In this paper, we have explored the potential of a framework based on association graphs for solving hypergraph isomorphism problems. Dynamics derived from the Baum-Eagon inequality have been introduced to optimize the objective function, thus finding the maximum clique in the association graphs that we have proven to be in one-to-one correspondence with the isomorphism. Impressive results have been obtained in terms of precision, as in 900 experiments run on random generated hypergraphs of different orders and connectivities we have always obtained 100% of correct isomorphisms, thus showing the great ability of the simple Baum-Eagon inequality to escape local minima in this kind of problems, and confirming earlier results on graphs [12,13]. From a computational

point of view, the exponentially increasing size of the association graph might become an issue for very sparse or very dense graphs, even though the use of exponential dynamics might ease this problem. In our future work we plan to use a regularized formulation introduced in [14], which has nicer theoretical properties than the one used in this paper, and also tackle the more challenging task of sub-hypergraph isomorphism.

## References

1. Barrow, H.G., Burstall, R.M.: Subgraph isomorphism, matching relational structures and maximal cliques. *Inf. Process. Lett.* **4**(4), 83–84 (1976)
2. Baum, L.E., Eagon, J.A.: An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bull. Am. Math. Soc.* **73**(3), 360–363 (1967)
3. Baum, L.E., Petrie, T., Soules, G., Weiss, N.: A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Stat.* **41**(1), 164–171 (1970)
4. Blakley, G.R.: Homogeneous nonnegative symmetric quadratic transformations. *Bull. Am. Math. Soc.* **70**(5), 712–715 (1964)
5. Duchenne, O., Bach, F., Kweon, I., Ponce, J.: A tensor-based algorithm for high-order graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(12), 2383–2395 (2011)
6. Garey, M.R., Johnson, D.S.: *Computers and Intractability*, vol. 29. WH Freeman, New York (2002)
7. Hou, J., Pelillo, M.: A game-theoretic hyper-graph matching algorithm. In: 24th International Conference on Pattern Recognition (ICPR) (2018)
8. Kozen, D.: A clique problem equivalent to graph isomorphism. *ACM SIGACT News* **10**(2), 50–52 (1978)
9. Lee, J., Cho, M., Lee, K.M.: Hyper-graph matching via reweighted random walks. *CVPR* **2011**, 1633–1640 (2011)
10. Motzkin, T.S., Straus, E.G.: Maxima for graphs and a new proof of a theorem of Turán. *Canad. J. Math.* **17**, 533–540 (1965)
11. Pelillo, M.: The dynamics of nonlinear relaxation labeling processes. *J. Math. Imaging Vis.* **7**(4), 309–323 (1997)
12. Pelillo, M.: A unifying framework for relational structure matching. In: *Proceedings of 14th International Conference on Pattern Recognition, (ICPR)*, pp. 1316–1319 (1998)
13. Pelillo, M.: Replicator equations, maximal cliques, and graph isomorphism. *Neural Comput.* **11**(8), 1933–1955 (1999)
14. Rota Bulò, S., Pelillo, M.: A generalization of the Motzkin-Straus theorem to hyper-graphs. *Optim. Lett.* **3**(2), 287–295 (2009)
15. Rota Bulò, S., Pelillo, M.: A game-theoretic approach to hypergraph clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(6), 1312–1327 (2013)
16. Yan, J., Zhang, C., Zha, H., Liu, W., Yang, X., Chu, S.M.: Discrete hyper-graph matching. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1520–1528 (2015)
17. Zhang, H., Ren, P.: Game theoretic hypergraph matching for multi-source image correspondences. *Pattern Recogn. Lett.* **87**, 87–95 (2017)



# Directed Network Analysis Using Transfer Entropy Component Analysis

Meihong Wu<sup>1</sup>, Yangbin Zeng<sup>1</sup>, Zhihong Zhang<sup>1</sup>(✉), Haiyun Hong<sup>1</sup>,  
Zhuobin Xu<sup>1</sup>, Lixin Cui<sup>2</sup>, Lu Bai<sup>2</sup>, and Edwin R. Hancock<sup>3</sup>

<sup>1</sup> Xiamen University, Xiamen, Fujian, China  
zhihong@xmu.edu.cn

<sup>2</sup> Central University of Finance and Economics, Beijing, China

<sup>3</sup> University of York, York, UK

**Abstract.** In this paper, we present a novel method for detecting directed network characteristics using histogram statistics based on degree distribution associated with transfer entropy. The proposed model in this paper established in information theory looks forward to learn the low dimensional representation of sample graphs, which can be obtained by transfer entropy component analysis (TECA) model. In particular, we apply transfer entropy to measure the transfer information between different time series data. For instances, for the fMRI time series data, we can use the transfer entropy to explore the connectivity between different brain functional regions effectively, which plays a significant role in diagnosing Alzheimers disease (AD) and its prodromal stage, mild cognitive impairment (MCI). With the properties of the directed graph in hand, we commence to further encode it into advanced representation of graphs based on the histogram statistics of degree distribution and multilinear principal component analysis (MPCA) technology. It not only reduces the memory space occupied by the huge transfer entropy matrix, but also enables the features to have a stronger representational capacity in the low-dimensional feature space. We conduct a classification experiment on the proposed model for the fMRI time series data. The experimental results verify that our model can significantly improve the diagnosis accuracy for MCI subjects.

**Keywords:** Transfer entropy · fMRI directed network  
Histogram statistic · Degree distribution

## 1 Introduction

Alzheimer's disease (AD) is an irreversible neurodegenerative disease. Mild cognitive impairment (MCI), a prodromal stage of AD, has gained much attention recently since MCI subjects tend to progress to clinical AD at an annual conversion rate of 10% to 15%, compared with normal controls (NC) who develop to AD at much lower annual conversion rate of approximately 1% to 2% [1]. FMRI [2, 3] is an imaging technique which can detect hemodynamic changes related to

neural activities based on the blood oxygenation level-dependent (BOLD) signals in grey matter (GM) regions. Graphs are powerful tools for representing complex patterns of interaction in high dimensional data [4]. The undirected brain functional network, generated by setting threshold in Pearson correlation coefficients matrix between pairs of brain regions, detect the interaction between different regions in [5]. Such undirected network ignores the causal relationship between the BOLD signal of different brain regions. On the contrary, in this work, we utilize directed graphs to depict the causal response between different brain functional regions, which can be indicative of the early onset of Alzheimer's disease.

Moreover, we turn to information theory and use entropy to define measures of graph characterizations. The von Neumann entropy was introduced by John von Neumann to measure irreversibility processes in quantum statistical mechanics [6]. Passerini and Severini [7] have shown how to use the von Neumann entropy to measure network irregularity. Since Shannon [8] introduces mutual information to measure the dependence between variables, [9] apply the mutual information in medical image processing and image registration task. Many researches make use of mutual information to quantify the overlap of the information content of two (sub)systems. However, the mutual information is symmetric, i.e., it makes no sense to analyze two (sub)systems that have a causal response. Therefore, we take advantage of transfer entropy which is asymmetric metrics to distinguish effectively driving and responding elements [10] and to detect asymmetry causal response between different brain functional regions.

The main contribution of this paper is threefold: first, by using the transfer entropy to depicting the causal response between different brain functional regions, our proposed TECA model can explore how the information flow in directed brain network, compared with the undirected graph method represented brain functional network [19]. Second, Based on the histogram statistics of degree distribution, we take huge transfer entropy matrix further enrichment into multi-dimensional histogram tensor, which not only reduces the memory space occupied by the redundant information, but also enables the features to possess a stronger representational capacity in the low-dimensional feature space. Finally, we conduct a classification experiment with the proposed model in the fMRI time series data, achieving significant improvements than other related methods.

The outline of this paper is as follows. Section 2 briefly reviews the preliminary concepts of information theory. Section 3 shows the synthetical framework of proposed TECA model and a step by step illustration on how representation of graph from original fMRI data can be constructed. The synthetical experimental evaluation will be presented in Sect. 4. Finally, Sect. 5 provides conclusions and directions for future work.

## 2 Preliminary Concepts

**Transfer Entropy.** Let us briefly review the important concepts of information theory. In the case of probability distribution  $p(i)$ , the average number of discrete variables with optimal coding independence is given by the Shannon entropy [13].

$$H = - \sum_i p_i \log_b p_i \quad (1)$$

where the sum extends over all state  $i$  and  $b$  is the base of logarithm. The mutual information of two variables  $X$  and  $Y$  with joint probability distribution  $p(x, y)$  can be regarded as the amount of information about another random variable contained in a random variable. The corresponding mutual information entropy is

$$I(X; Y) = \sum_x \sum_y p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \quad (2)$$

Note that symmetry is one of the characteristics of mutual information, i.e.,  $I(X; Y) = I(Y; X)$ , so it would be inappropriate to measure the causal response between two (sub)systems using mutual information. On the contrary, transfer entropy is able to distinguish effectively driving and responding elements and to detect asymmetry in different subsystems [10]. In the absence of information flow from  $X$  to  $Y$ , the state of  $X$  has no effect on the transition probability of the system  $Y$ , i.e., transfer entropy is the information that is required to predict the state of the system  $Y$  in the case of  $p(y_{n+1}|y_n^{(k)}, x_n^{(l)})$  rather than  $p(y_{n+1}|y_n^{(k)})$ , which can be defined as follow:

$$T_{X \rightarrow Y} = \sum p(y_{n+1}, y_n^{(k)}, x_n^{(l)}) \log \frac{p(y_{n+1}|y_n^{(k)}, x_n^{(l)})}{p(y_{n+1}|y_n^{(k)})} \quad (3)$$

Generally, the most common choices are  $l = k$  or  $l = 1$  and note that transfer entropy is asymmetric, i.e.,  $T_{X \rightarrow Y} \neq T_{Y \rightarrow X}$ .

**Multilinear Principal Component Analysis.** First of all, let us introduce the concept of tensor, which denotes a multi-dimensional matrix and the position of the elements of which are to be determined by the indices that needs two more [14]. We utilize tensor object to represent directed graph in the high-dimensional Euclidean space (more details will be provided below). Facing with such a high dimensional tensor, which is difficult to directly extract effective feature information to fit the distribution of samples. On the other hand, the straightforward application of principal component analysis (PCA) to tensor object requires its reconstruction of the vector of high dimension, which obviously leads to the calculation of high processing cost and the increase of memory demand. To overcome this challenge, multilinear principal component analysis (MPCA) method was first proposed by [11], which performs feature reduction by exploring a multilinear projection matrix that retains most of the original information of input tensor object.

Formally, let  $\mathcal{X} \in \mathcal{R}^{I_1 \times I_2 \times \dots \times I_N}$  denotes the tensor object, where  $I_n$  is the  $n$ -mode dimension in tensor space. The multilinear transformation  $\{U^{(n)} \in \mathcal{R}^{I_n \times P_n}, n = 1, \dots, N\}$  projects the original tensor space  $\mathcal{R}^{I_1 \times I_2 \times \dots \times I_N}$  into the

subspace  $\mathcal{R}^{P_1 \times P_2 \times \dots \times P_N}$ , where  $P_n < I_n, n = 1, \dots, N$ . With a set of multilinear transformation matrix  $U$  in hand, the projection of  $\mathcal{X}$  in the tensor subspace  $\mathcal{R}^{P_1 \times P_2 \times \dots \times P_N}$  is computed by

$$\mathcal{Y} = \mathcal{X} \times U^{(1)T} \times \dots \times U^{(N)T} \tag{4}$$

Note that  $\mathcal{Y} \in \mathcal{R}^{P_1 \times P_2 \times \dots \times P_N}, P_n < I_n, n = 1, \dots, N$ , so that the amount of elements in the subspace less than those in the original space, i.e., the low dimensional representation of the tensor can be constructed.

### 3 Transfer Entropy Component Analysis Model

In this section, we first show how the transfer entropy encodes fMRI time series data. Next, we integrate the global property of the directed graph into the advanced multi-dimensional matrix based on the histogram statistics of degree distribution. Once the combined the idea from MPCA, the representation of graph in low dimensional Euclidean space can be constructed. The Fig. 1 shows the framework of proposed TECA model.

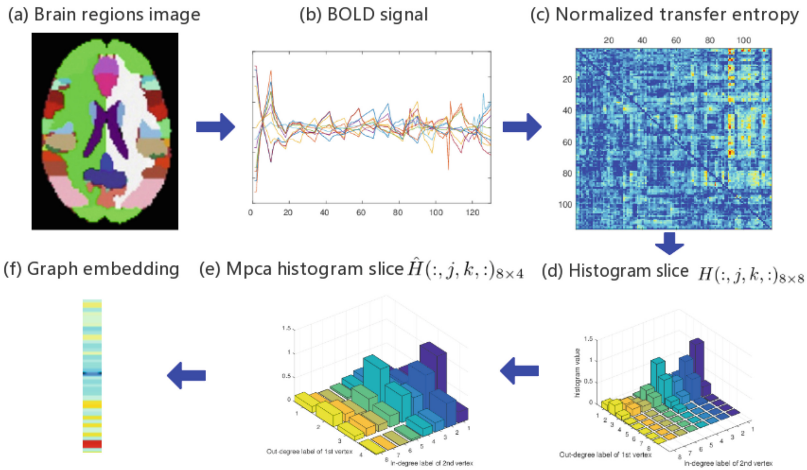


Fig. 1. The framework of TECA model.

**Transfer Entropy of Sample Graphs.** From the definition of the previous section, transfer entropy is a quantitative measure of information transfer between two dynamic processes  $X$  and  $Y$ . For fMRI time series data, which contains BOLD signals from different brain regions at the same time series. With the transfer entropy to hand, we commence to compute the degree of causal response between different brain functional regions through Eq. (3). Formally,

$\mathcal{T} = \{T^1, \dots, T^m\}$  denotes the set of transfer entropy matrix, where  $T^i$  can be computed by

$$T_{xy}^i = \begin{cases} T_{X \rightarrow Y}, & \text{if } x \neq y \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

The reason why the diagonal element is zero is that the transfer entropy is the causal correlation between different time series, and it can't work in the same (sub)system.

Due to the loss of the equipment acquisition signal or the instability of the data, the transfer entropy matrix usually contains much noise. Recently, there have been some literature to eliminate the noise of transfer entropy by filtering or subtracting the average from  $X$  to  $Y$  using shuffled  $X$  repeat several times [15, 16]. In this case, the normalized transfer entropy matrix is given by

$$NT_{xy}^i = \begin{cases} \frac{T_{X \rightarrow Y} - \langle T_{X_{shuffled} \rightarrow Y} \rangle}{H(Y_{n+1}|Y_n)}, & \text{if } x \neq y \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

where the denominator denotes the conditional entropy of time series  $Y$  at time  $n + 1$  given its value at time  $n$ , which is given by

$$H(Y_{n+1}|Y_n) = - \sum p(Y_n + 1, Y_n) \log \frac{p(Y_n + 1, Y_n)}{p(Y_n)} \tag{7}$$

Note that  $NT^i$  is an asymmetric transfer entropy matrix whose elements are non-binary. In other words, we can directly regard this matrix as the weight matrix of the sample graph. To our knowledge, a hypothesis is proposed by [17] that the node pairs disconnected in the real network may have potential connectivity, rather than either connected or disconnected. Therefore, if the weight matrix is constructed by setting the threshold in asymmetric transfer entropy matrix, such weight matrix may lose the original information of graph.

**Histogram Statistics Based on Degree Distribution.** With the transfer entropy matrix to hand, in this subsection, we aim to further integrate the global property in the huge transfer matrix into a more efficient histogram matrix based on the histogram statistics of degree distribution. More specifically, let  $\mathcal{G} = \{G_1, \dots, G_n\}$  denotes the set of sample graphs, where  $G_i = (V_i, E_i)$  is the sample graph  $G_i$ . For a sample graph  $G_i$ , we assume that  $G_i = (V_i, E_i)$  is a directed graph without loss of generality, then the weight matrix  $D^i = NT^i$ . With the weight matrix to hand, we can define the in-degree and out-degree of node  $a$  as follow:

$$d_a^{(i),in} = \sum_{b \in V_i} D_{ba}^i, \quad d_a^{(i),out} = \sum_{b \in V_i} D_{ab}^i. \tag{8}$$

Note that the in-degree and out-degree of node are floating point numbers, which is consistent with the previous assumption that the node pairs that are not directly connected by edge in the real network may have potential connectivity.

To capture the global property of the directed graph, the histogram statistics based on the degree distribution is proposed by [18], which first construct a four-dimensional tensor object  $H \in \mathcal{R}^{\beta \times \beta \times \beta \times \beta}$  whose elements represent the histogram bin-contents and indices represent the degree label of the nodes. For instance,  $H_{1234}$  is the entropy contribution from out-degree 1 and in-degree 2 of nodes, pointing to nodes with out-degree 3 and in-degree 4. In our work, similarly, we directly utilize histogram statistics based on degree distribution to calculate the transfer entropy contribution from different degree nodes. The element of transfer entropy histogram matrix  $H^i$  of directed graph  $G_i$  is formally given as

$$H^i_{ojul} = \sum_{\substack{d_a^{out}=o, d_a^{in}=j, \\ d_b^{out}=u, d_b^{in}=l}} \{D_{ab}^i \times NT_{ab}^i\}. \tag{9}$$

where  $o, j, u, l = 1, \dots, \beta$  and  $\beta$  is the number of bin.

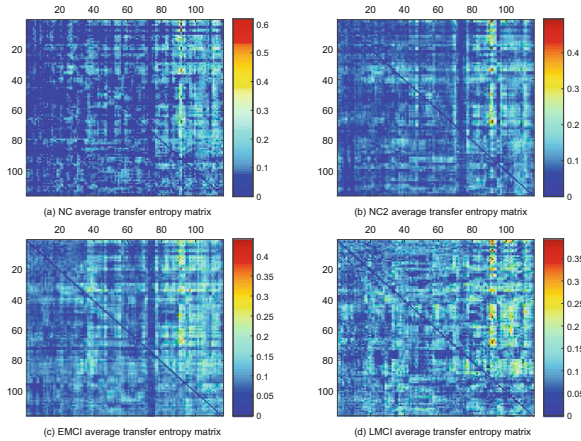
**Graph Embedding via MPCA.** The main subject in this subsection is to further integrate the global property in the multi-dimensional histogram matrix into a low dimensional representation of graph via feature reduction algorithm of multilinear principal component analysis, which can greatly reduce the memory space occupied by redundant information and enhance the representation of sample graphs. For instances, for a directed graph  $G_i$  whose histogram tensor is four dimensional object  $H^i \in \mathcal{R}^{I_1 \times I_2 \times I_3 \times I_4}$ , a set of multilinear transformation matrix  $U = \{U^{(1)}, \dots, U^{(4)}\}$  can be computed by MPCA method. According to Eqs. (4, 9), the low dimensional tensor object  $\tilde{H}^i \in \mathcal{R}^{P_1 \times P_2 \times P_3 \times P_4}$  can be generated. With the low dimensional tensor in hand, we concatenate each element of the tensor object to generate the representation of graph.

## 4 Experiment

**Dataset.** In the previous section, we have a general knowledge of fMRI dataset and Alzheimer’s disease. Next, we introduce the data formats and features of the dataset detailedly so that relevant experiments can be carried out. The subjects of fMRI dataset can be divided into four categories according to the severity of the disease, namely Healthy Control (NC), Healthy Control2 (NC2), Early Mild Cognitive Impairment (EMCI) and Late Mild Cognitive Impairment (LMCI). Particularly, there are 43 subjects in NC group, 17 in NC2, 16 in EMCI and 38 in LMCI group, each of subject consisted of 116 brain functional regions.

**Experiment Result and Discussion.** Now we exhibit the application of the TECA model to investigate the effectiveness in the fMRI dataset. We first calculate the transfer entropy matrix of the sample graph according to the Eq. (3) and present the average normalized transfer entropy matrix with different subjects in Fig. 2. Although the numerical scale of transfer entropy is tiny, we still have a observation that the transfer entropy matrix of EMCI and LMCI are brighter

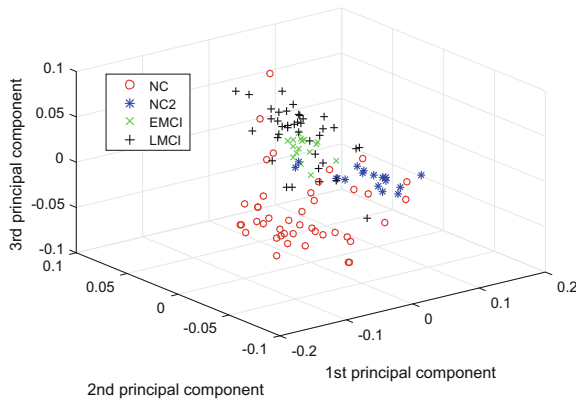




**Fig. 2.** The average normalized transfer entropy (NT) matrix of four different groups.

than those of NC and NC2 in the right half part of the matrix. In other word, the causal response of different brain functional regions, to some extent, can be detected by transfer entropy.

A low dimensional representation of graph can be constructed by histogram statistics based on degree distribution and MPCA algorithm. Figure 3 shows the results of mapping the graphs into a 3-dimensional feature space represented by the first three principal components of graph embedding. From this figure, there is a straightforward observation that different subjects almost can be divided, which performs the discrimination of graph embedding.



**Fig. 3.** Multilinear principal component analysis performance of four categories based on transfer entropy.

To compare with other related methods at a more accurate degree, we not only place our proposed method on binary classification task in fMRI dataset, which put the EMCI and LMCI group in one category named MCI and the rest of groups in the other category named NC, but also use the same evaluation metrics as [19]. And we compare with the dComb method proposed in [19], which combines the dynamic functional correlation tensors with the dynamic functional connectivity in grey matter to classify the fMRI data. For dComb method, we directly use the implementation provided by its author. For our method, the hyper-parameters  $\beta$  (the number of bin) and the embedding dimension  $k$  are tuned by using grid search on the validation set. And the training ratio on the fMRI dataset is increased from 10% to 90%. Beside, due to limited samples, the ten-fold cross validation of livsvm classifier [12] is applied to select the appropriate parameters and guarantee the reliability of the classifier.

**Table 1.** Performance of different methods in MCI classification

Method	Accuracy	Sensitivity	Specificity	AUC	F-score
dComb	78.70	77.78	79.63	0.8449	78.50
TECA	82.53	81.15	83.37	0.8754	81.86
T-TECA	80.30	79.25	79.70	0.8423	79.95
N-TECA	<b>85.51</b>	<b>86.80</b>	<b>85.30</b>	<b>0.9122</b>	<b>85.65</b>

Table 1 summaries the results with different evaluation metrics on fMRI dataset, numbers in bold present the highest performance in reach column. The one contains normalized the transfer entropy matrix named N-TECA model and the non-normalized method is TECA model, and T-TECA one constructs weight matrix by setting the threshold, which selected by using grid search on the validation set. From Table 1, we have the following observation: our proposed model achieves significant improvements than other related methods on fMRI dataset, which demonstrates the effectiveness on detecting the causal response between different brain functional regions based on transfer entropy.

There are two crucial hyper-parameters in TECA, i.e.,  $\beta$  and  $k$ . For  $\beta$ , it determines the number of histogram bin, which directly affects the magnitude of histogram and the capacity of mapping the property of graph. We determine the value of  $\beta$  according to the classification accuracy in validation set. In the left part of Fig. 4, we present the classification accuracy of validation set over different settings of the number of bin  $\beta$  and  $k$  to 5 on fMRI data. From this figure, we have the observation that classification accuracy becomes stable with the training ratio  $\beta$  growth and setting the number of bin  $\beta$  to 15 possesses best performance. The hyper-parameter  $k$  controls the dimension of feature space and the performance of the classifier. We show the classification accuracy in validation set in the right part of Fig. 4 when  $k$  take values under different orders of magnitude and  $\beta$  setted to 15. From this figure, we can observe the phenomenon that the performance is best with setting the dimension of graph

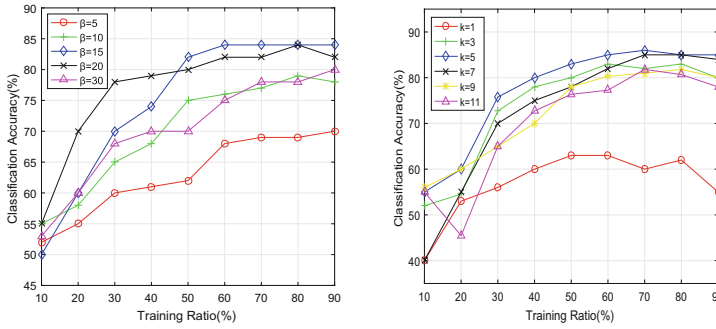


Fig. 4. Parameter sensitivity.

embedding to 5. However, when  $k$  is too large, the performance decreases too. The reason is that, in this case, too many feature may make classifier poor while the feature space is too large, i.e., the number of sample is relatively small and the classifier possesses poor understand on distribution of data.

## 5 Conclusion

In this paper, we present a novel method for detecting network characteristics using histogram statistics based on degree distribution associated with transfer entropy. The proposed TECA model is to explore the causal relationship between different (sub)systems. To this end, we commence to construct the transfer entropy matrix of sample graphs, measuring the transfer information between different brain functional regions. With the global properties of the directed graph in hand, a low dimensional representation of graph can be generated by histogram statistics based on degree distribution and multilinear principal component analysis method. Experimental results reveal that the proposed TECA model possess the significant improvement on graph classification with dynamic time series data. Besides, further work maybe focus on how to learn generative model to detect the structure of directed network based on transfer entropy, e.g., generative supergraph model.

## References

1. Lo, R.Y., et al.: Longitudinal change of biomarkers in cognitive decline. *Arch. Neurol.* **68**(10), 1257–1266 (2011)
2. Machulda, M.M., et al.: Functional MRI changes in amnesic and non-amnesic MCI during encoding and recognition tasks (2009)
3. Wee, C.Y., Yang, S., Yap, P.T., Shen, D.: Sparse temporally dynamic resting-state functional connectivity networks for early MCI identification. *Brain Imaging Behav.* **10**(2), 342–356 (2016)
4. Kang, U., Tong, H., Sun, J.: Fast random walk graph kernel (2012)

5. Onias, H., et al.: Brain complex network analysis by means of resting state fMRI and graph analysis: will it be helpful in clinical epilepsy? *Epilepsy Behav.* **38**, 71–80 (2014)
6. Edwards, D.A.: *The mathematical foundations of quantum mechanics* (1955)
7. Passerini, F., Severini, S.: The von Neumann entropy of networks. *SSRN Electron. J.* (12538) (2008)
8. Shannon, C.E.: A mathematical theory of communication. *Bell Syst. Tech. J.* **27**(4), 379–423 (1948)
9. Pluim, J.P.W., Maintz, J.B.A., Viergever, M.A.: Image registration by maximization of combined mutual information and gradient information. *IEEE Trans. Med. Imaging* **19**(8), 809–814 (2000)
10. Schreiber, T.: Measuring information transfer. *Phys. Rev. Lett.* **85**(2), 461–464 (2000)
11. Haiping, L., Plataniotis, K.N., Venetsanopoulos, A.N.: MPCA: multilinear principal component analysis of tensor objects. *IEEE Trans. Neural Netw.* **19**(1), 18 (2008)
12. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2**(3), 1–27 (2011)
13. Shannon, C.E.: *The mathematical theory of communication*, 1963. *MD Comput.* **14**(4), 306 (1997)
14. De Lathauwer, L., De Moor, B., Vandewalle, J.: On the best rank-1 and rank- $(r_1, r_2, \dots, r_n)$  approximation of higher-order tensors. *SIAM J. Matrix Anal. Appl.* **21**(4), 1324–1342 (2000)
15. Neymotin, S.A., Jacobs, K.M., Fenton, A.A., Lytton, W.W.: Synaptic information transfer in computer models of neocortical columns. *J. Comput. Neurosci.* **30**(1), 69–84 (2011)
16. Gourvitch, B., Eggermont, J.J.: Evaluating information transfer between auditory cortical neurons. *J. Neurophysiol.* **97**(3), 2533 (2008)
17. Martin, T., Ball, B., Newman, M.E.: Structural inference for uncertain networks. *Phys. Rev. E Stat. Nonlinear Soft Matter Phys.* **93**(1–1), 012306 (2016)
18. Ye, C., Wilson, R.C., Hancock, E.R.: Network analysis using entropy component analysis. *IMA J. Complex Netw.* (2017)
19. Chen, X., Zhang, H., Zhang, L., Shen, C., Lee, S.W., Shen, D.: Extraction of dynamic functional connectivity from brain grey matter and white matter for MCI classification. *Hum. Brain Mapp.* **38**(10), 5019 (2017)



# A Mixed Entropy Local-Global Reproducing Kernel for Attributed Graphs

Lixin Cui<sup>1</sup>, Lu Bai<sup>1(✉)</sup>, Luca Rossi<sup>2</sup>, Zhihong Zhang<sup>3</sup>, Lixiang Xu<sup>4</sup>,  
and Edwin R. Hancock<sup>5</sup>

<sup>1</sup> Central University of Finance and Economics, Beijing, China  
bailucs@cufe.edu.cn

<sup>2</sup> Aston University, Birmingham, UK

<sup>3</sup> Xiamen University, Xiamen, Fujian, China

<sup>4</sup> Hefei University, Hefei, Anhui, China

<sup>5</sup> University of York, York, UK

**Abstract.** In this paper, we develop a new mixed entropy local-global reproducing kernel for vertex attributed graphs based on depth-based representations that naturally reflect both local and global entropy based graph characteristics. Specifically, for a pair of graphs, we commence by computing the nest depth-based representations rooted at the centroid vertices. The resulting mixed local-global reproducing kernel for a pair of graphs is computed by measuring a basic  $H^1$ -reproducing kernel between their nest representations associated with different entropy measures. We show that the proposed kernel not only reflect both the local and global graph characteristics through the nest depth-based representations, but also reflect rich edge connection information and vertex label information through different kinds of entropy measures. Moreover, since both the required basic  $H^1$ -reproducing kernel and the nest depth-based representation can be computed in a polynomial time, the new proposed kernel processes efficient computational complexity. Experiments on standard graph datasets demonstrate the effectiveness and efficiency of the proposed kernel.

**Keywords:** Local-global graph kernels · Attributed graphs · Entropy

## 1 Introduction

In machine learning and pattern recognition, graph kernels are powerful tools for analyzing graph-based data [14]. Comparing to classical graph embedding methods that approximate graphs into vectors [14], graph kernels not only provide a way of applying standard machine learning techniques (e.g., SVM, kPCA, etc.) to graph datasets, but also better preserve structural information in a high dimensional Hilbert space [13].

Generally speaking, most existing state-of-the-art graph kernels fall into instances of the R-convolution kernel. The R-convolution is a generic way of defining graph kernels based on the idea of decomposing graphs into substructures and comparing pairs of specific substructures. Under this scenario, most graph kernels based on R-convolution can be categorized into three classes, i.e., the graph kernels based on counting pairs of isomorphic (a) walks [16], (b) paths [1], and (c) restricted subgraphs or subtree substructures [8]. One main drawback arising in these R-convolution kernels is that they only reflect restricted topological information of graph structures with limited sized substructures. As a result, the R-convolution kernels fail to reflect global graph characteristics.

To overcome the restriction on local graph characteristics of existing R-convolution kernels, a family of graph kernels that are based on global graph characteristics have been developed. For instance, Johansson et al. [11] have developed a Lovász kernel that uses the Lovász number and its associated orthonormal representation to capture global graph characteristics. Xu et al. [18] have proposed a local-global mixed reproducing kernel based on the approximate von Neumann entropy through the adjacency matrix. Bai et al. and Rossi et al. [6, 15] have developed a family of quantum graph kernels based on the quantum Jensen-Shannon divergence associated with quantum walk entropies. Specifically, these kernels capture the global graph characteristics by evolving the quantum walk to probe the whole graph structures. Furthermore, to develop a graph kernel that can simultaneously capture both the local and global graph characteristics, Bai et al. [5] have developed a local-global graph kernel through the dynamic time warping framework. Specifically, they commence by computing a nest representation for each graph that gradually lead a centroid vertex (local characteristics) to the global graph structure (the global characteristics). For a pair of graphs, the resulting local-global graph kernel is defined by measuring a dynamic time warping inspired kernel between their individual nest representations. Unfortunately, all the aforementioned local, global, and local-global graph kernels cannot accommodate graph vertex labels and thus are restricted to only un-attributed graphs.

The aim of this paper is to further develop a new local-global graph kernel, namely the mixed entropy local-global reproducing kernel, for vertex attributed graphs. Specifically, for each graph, we commence by decomposing the graph structure into a family of  $K$ -layer expansion subgraphs with increasing layers. Unlike the previous work [5] that only employs the Shannon entropy measure, we compute three nest depth-based representations for each graph by measuring the approximated von Neumann entropy [9], the Shannon entropy associated with steady state random walks [3], and the label Shannon entropy associated with vertex labels on the family of expansion subgraphs, respectively. We show that these nest representations can naturally reflect both local and global characteristics in terms of the expansion subgraphs with increasing layers. The resulting mixed local-global reproducing kernel for a pair of vertex attributed graphs is computed by measuring a basic  $H^1$ -reproducing kernel between their nest depth-based representations associated with different entropy measures.

The proposed kernel cannot only simultaneously capture the local and global entropy-based graph characteristics through the nest depth-based representations, but also reflect comprehensive edge connection information and vertex label information through the different kinds of entropy measures. Moreover, since both the required basic  $H^1$ -reproducing kernel and the nest depth-based representation can be computed in a polynomial time, the new proposed kernel processes efficient computational complexity. Experiments on standard graph datasets demonstrate the effectiveness and efficiency of the proposed kernel.

The remainder of this paper is organized as follows. Section 2 reviews the preliminary concepts that will be used in this work. Section 3 defines the proposed mixed entropy local-global reproducing kernel. Section 4 provides the experimental evaluation. Section 5 concludes this work.

## 2 Preliminary Concepts

In this section, we introduce some preliminary concepts that will be used in this work. We commence by introducing a new reproducing kernel that is an extension of the  $H^1$ -reproducing kernel to the graph kernel realm. Moreover, we introduce three kinds of graph entropy measure method. Finally, we review the concept of the nest depth-based representations of graphs associated with different entropy measures.

### 2.1 Reproducing Kernels

A Hilbert Space is an inner product space that is complete and separable with respect to the norm defined by the inner product. A Hilbert space of complex-valued functions which possesses a reproducing kernel is called a RKHS or a proper Hilbert space. RKHS is the space of functions with the nice property that if a function  $f(x)$  is close to a function  $g(x)$  in the sense of the distance derived from the inner product.

**Definition 1 (The reproducing kernel).** A function:  $K : E \times E \rightarrow C$ ,  $(s, t) \mapsto K(s, t)$  is a reproducing kernel of the Hilbert space  $H$  if and only if

- (i)  $\forall t \in E, K(., t) \in H$ ;
- (ii)  $\forall t \in E, \forall \phi \in H \langle \phi, K(., t) \rangle = \phi(t)$ .

The last condition (ii) is called the reproducing property: the value of the function  $\phi$  at the point  $t$  is reproduced by the inner product of  $\phi$  with  $K(., t)$ .  $\square$

In this subsection, we review how to use the  $H^1$ -reproducing kernel to define a basic reproducing kernel [17]. We start with the concept of the  $H^1$ -reproducing kernel, which can be seen as an extension of the  $H^1$ -reproducing kernel to the graph kernel realm. Specifically, in the following Lemma 1, we obtain the basic solution of the generalized differential operator using the Delta function [18, 19]. The Delta function  $\sigma(x)$  physically represents the density of an idealized point

mass or a point charge. In practice, the Delta function plays an important role in partial differential equations, mathematical physics, Fourier analysis, and theory of probability [2].

**Lemma 1.** *Let  $K_1(x)$  be the basic solution of the operator  $L = 1 - \frac{d^2}{dx^2}$ , then the basic reproducing kernel of  $H^1(R)$  is  $K_1(x - y)$ .*

By [18], we know the function

$$K_1(x, y) = K_1(x - y) = \frac{1}{2}e^{-|x-y|}, \tag{1}$$

which obviously satisfies condition (i) and (ii) of Definition 1. So  $K_1(x, y) = K_1(x - y)$  is a  $H^1$ -reproducing kernel in  $H^1(R)$ .  $\square$

Intuitively, the basic reproducing kernel  $K_1$  allows one to define a new basic reproducing graph kernel associated with any type of graph characteristics values, e.g., the graph entropy measures suggested in [4]. Moreover, the computation of the basic reproducing kernel  $K_1$  only requires time complexity  $O(1)$  and the computation of some entropy measures are only quadratic in the number of vertices. Therefore, the basic reproducing kernel  $K_1$  provides us a way of defining new fast graph kernel associated with graph entropy measures. For instance, [17] have proposed a hybrid reproducing kernel by measuring the basic reproducing kernel  $K_1$  between the entropies of global graphs. Since the associated entropy measures only require time complexity  $O(n^2)$  where  $n$  is the vertex number of the graph, their hybrid reproducing kernel only requires time complexity  $O(n^2 + 1)$ . Unfortunately, the hybrid reproducing kernel between global graph entropies neither reflects local characteristics from the global graph structure, nor accommodates vertex labels for attributed graphs.

## 2.2 Entropy Measures for Graphs

We review the concepts of three graph entropy measures, namely the approximate von Neumann entropy [10], the Shannon entropy associated with steady state random walks [4], and the label Shannon entropy associated with vertex labels. Let a graph be denoted as  $G(V, E)$ , where  $V$  is the vertex set and  $E \subseteq V \times V$  is the undirected edge set. The adjacency matrix  $A$  of the graph  $G(V, E)$  is a  $|V| \times |V|$  symmetric matrix and each element satisfies

$$A(i, j) = \begin{cases} 1 & \text{if } (v_i, v_j) \in E; \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The vertex degree matrix  $D$  of  $G$  is a diagonal matrix whose elements are defined by

$$D(v_i, v_i) = d(i) = \sum_{v_j \in V} A(i, j). \tag{3}$$



**Definition 2 (The Approximate Von Neumann Entropy).** Based on the definition in [10], we can compute an approximate von Neumann entropy for the graph  $G(V, E)$  in terms of its degree matrix  $D$  as

$$H_{VN}(G) = 1 - \frac{1}{|V|} - \sum_{(v_i, v_j) \in E} \frac{1}{|V|^2 d(i)d(j)}, \tag{4}$$

where each edge  $(v_i, v_j) \in E$  is indicated by the adjacency matrix  $A$ . □

**Definition 3 (The Shannon Entropy).** For each vertex  $v_i \in V$  of the graph  $G(V, E)$ , the probability of a steady state random walk on  $G(V, E)$  visiting  $v_i$  is

$$P(i) = d(i) / \sum_{v_j \in V} d(j). \tag{5}$$

From this probability distribution  $P$ , we can straightforwardly compute the Shannon entropy as

$$H_S(G) = - \sum_{i=1}^{|V|} P(i) \log P(i). \tag{6}$$

□

Both the aforementioned entropy measures require computational complexity  $O(n^2)$  ( $n$  is the vertex number), since their required vertex degree statistics are computed based on the  $n^2$  elements of the graph adjacency matrix. Moreover, both the entropy measures can reflect rich edge connectivity information of the graphs in terms of the vertex degrees. Unfortunately, neither of the entropy measure can accommodate attributed graphs. To overcome this problem, we introduce a new label Shannon entropy.

**Definition 4 (The Label Shannon Entropy).** Let  $\mathcal{L} = \{l_1, \dots, l_x, \dots, l_{|\mathcal{L}|}\}$  be the vertex label set of graph  $G(V, E)$ . We commence by reviewing the labels of all vertices, and compute the frequency of each particular label  $l_x$  contained in  $G(V, E)$  as  $c(l_x)$ . The probability  $P_L(l_x)$  of each label  $l_x$  is

$$P_L(l_x) = \frac{c(l_x)}{\sum_{x=1}^{|\mathcal{L}|} c(l_x)}. \tag{7}$$

From the label probability distribution  $P_L$ , we compute a label Shannon entropy as

$$H_{LS}(G) = - \sum_{x=1}^{|\mathcal{L}|} P_L(l_x) \log P_L(l_x). \tag{8}$$

□

Similar to the approximated von Neumann entropy defined in Eq. 4 and the random walk Shannon entropy defined in Eq. 6, the label Shannon entropy for a graph also require time complexity  $O(n^2)$ , where  $n$  is the vertex number. This is because for each vertex label  $l_x$  we need to review the labels of the remaining  $n - 1$  vertices, and the number of the different vertex label is at most  $n$ . The label Shannon entropy can directly accommodate vertex label by exploring the label frequency.

### 2.3 Centroid Depth-Based Complexity Traces

In this subsection, we review the concept of the nest centroid depth-based representation [4]. Assume a graph  $G(V, E)$  where  $V$  and  $E$  are the vertex and edge sets respectively. We commence by computing the shortest path matrix  $S_G$  based on Dijkstra’s algorithm. Specifically, each element  $S_G(v, u)$  of  $S_G$  represents the shortest path length between vertices  $v \in V$  and  $u \in V$ . Assume  $S(v)$  is the average length of all shortest paths from  $v \in V$  to the remaining vertices, i.e.,  $S(v) = \frac{1}{|V|} \sum_{u \in V} S_G(v, u)$ . Based on [4], the index of the centroid vertex  $\hat{v}_C$  of  $G$  can be identified by

$$\hat{v}_C = \arg \min_v \sum_{u \in V} [S_G(v, u) - S_V(v)]^2. \tag{9}$$

Let  $N_{\hat{v}_C}^K$  be a vertex subset of  $G(V, E)$  satisfying  $N_{\hat{v}_C}^K = \{u \in V \mid S_G(\hat{v}_C, u) \leq K\}$ . For  $G(V, E)$ , we construct a family of  $K$ -layer expansion subgraphs  $\mathcal{G}_K(\mathcal{V}_K; \mathcal{E}_K)$  rooted at its centroid vertex  $\hat{v}_C$  as

$$\begin{cases} \mathcal{V}_K = \{u \in N_{\hat{v}_C}^K\}; \\ \mathcal{E}_K = \{(u, v) \subset N_{\hat{v}_C}^K \times N_{\hat{v}_C}^K \mid (u, v) \in E\}. \end{cases} \tag{10}$$

Note that the number of the expansion subgraphs is equal to the greatest length  $L$  of the shortest paths from the centroid vertex to the remaining vertices. Moreover, the  $L$ -layer expansion subgraph is the global structure of  $G(V, E)$ .

**Definition 5 (Centroid Depth-based Representation).** For a graph  $G(V, E)$  and its family of centroid expansion subgraphs  $\{\mathcal{G}_1, \dots, \mathcal{G}_K, \dots, \mathcal{G}_L\}$ . The centroid depth-based representation  $\text{DB}(G)$  of  $G$  is defined as

$$\text{DB}(G) = \{H(\mathcal{G}_1), \dots, H(\mathcal{G}_K), \dots, H(\mathcal{G}_L)\}, \tag{11}$$

where  $H(\mathcal{G}_K)$  can be any of the approximated von Neumann entropy defined in Eq. 4, the random walk Shannon entropy defined in Eq. 6, or the label Shannon entropy defined in Eq. 8.  $\square$

Bai et al. [5] have indicated that the centroid depth-based representation  $\text{DB}(G) = \{H(\mathcal{G}_1), \dots, H(\mathcal{G}_K), \dots, H(\mathcal{G}_L)\}$  of each graph  $G$  preserves **nest property**, i.e., the entropy-based information of each  $K$ -layer expansion subgraph encapsulates that of the 1-layer to  $K - 1$ -layer expansion subgraphs. As a result, the centroid depth-based representation  $\text{DB}(G)$  gradually leads the entropy measures from the local centroid vertex to the global graph structure, and  $\text{DB}(G)$  can be seen as a **nest depth-based representation** that simultaneously reflects both the local and global structure information of  $G$ .  $\text{DB}(G)$  provides a way of developing new local-global kernels for graphs.

## 3 The Mixed Entropy Local-Global Reproducing Kernel

### 3.1 A Nest Aligned Kernel from the Dynamic Time Warping Framework

Let  $G_P(V_P, E_P)$  and  $G_Q(V_Q, E_Q)$  be a pair of sample graphs from a graph set  $\mathbf{G}$ . We commence by computing the nest depth-based representations of  $G_P$  and  $G_Q$  as

$$\text{DB}(G_P) = \{H(\mathcal{G}_{P;1}), \dots, H(\mathcal{G}_{P;K}), \dots, H(\mathcal{G}_{P;L^{\max}})\}$$

and

$$\text{DB}(G_Q) = \{H(\mathcal{G}_{Q;1}), \dots, H(\mathcal{G}_{Q;K}), \dots, H(\mathcal{G}_{Q;L^{\max}})\},$$

respectively, where  $\mathcal{G}_{P;K}$  and  $\mathcal{G}_{Q;K}$  are the  $K$ -layer expansion subgraphs rooted at the centroid vertices of  $G_P$  and  $G_Q$ , and  $L^{\max}$  is the greatest length of the shortest paths rooted at the centroid vertices over all graphs in  $\mathbf{G}$ . Based on the basic  $H^1$ -reproducing kernel  $K_1$  defined in Sect. 2.1, we develop a nest reproducing graph kernel  $k_{\text{NR}}$  between  $G_P$  and  $G_Q$  as

$$\begin{aligned} k_{\text{NR}}(G_P, G_Q) &= k_{\text{NR}}\{\text{DB}(G_P), \text{DB}(G_Q)\} \\ &= \sum_{K=1}^{L^{\max}} K_1\{H(\mathcal{G}_{P;K}), H(\mathcal{G}_{Q;K})\} \\ &= \frac{1}{2} \sum_{K=1}^{L^{\max}} e^{-|H(\mathcal{G}_{P;K}) - H(\mathcal{G}_{Q;K})|}. \end{aligned} \quad (12)$$

When we associate the approximated von Neumann entropy  $H_V$  defined in Eq. 4, the random walk Shannon entropy  $H_S$  defined in Eq. 6, and the label Shannon entropy  $H_{LS}$  defined in Eq. 8 with the nest reproducing graph kernel  $k_{\text{NR}}(G_P, G_Q)$ , we define a new mixed entropy local-global graph kernel  $k_{\text{MELG}}$  between  $G_P$  and  $G_Q$  as

$$\begin{aligned} k_{\text{MELG}}(G_P, G_Q) &= k_{\text{NR}}^V(G_P, G_Q) + k_{\text{NR}}^S(G_P, G_Q) + k_{\text{NR}}^{LS}(G_P, G_Q) \\ &= \frac{1}{2} \sum_{K=1}^{L^{\max}} e^{-|H_V(\mathcal{G}_{P;K}) - H_V(\mathcal{G}_{Q;K})|} + \frac{1}{2} \sum_{K=1}^{L^{\max}} e^{-|H_S(\mathcal{G}_{P;K}) - H_S(\mathcal{G}_{Q;K})|} \\ &\quad + \frac{1}{2} \sum_{K=1}^{L^{\max}} e^{-|H_{LS}(\mathcal{G}_{P;K}) - H_{LS}(\mathcal{G}_{Q;K})|}. \end{aligned} \quad (13)$$

Intuitively, the proposed kernel  $k_{\text{MELG}}$  is positive definite (**pd**), since  $k_{\text{MELG}}$  is based on the sum of the basic (**pd**)  $H^1$ -producing kernel  $K_1$ .  $k_{\text{MELG}}$  can simultaneously reflect the local and global graph characteristics in terms of the nest depth-based representation. Moreover,  $k_{\text{MELG}}$  not only reflect rich edge connection information through the approximate von Neumann entropy and the random walk Shannon entropy, but also accommodate the vertex label information through the label Shannon entropy measure.

Finally, note that, the proposed mixed entropy local-global kernel  $k_{\text{MELG}}$  is related to the hybrid reproducing kernel developed by [17], since both of the kernels are based on the basic  $H^1$ -reproducing kernel  $K_1$ . However, the proposed kernel  $k_{\text{MELG}}$  is still theoretically different from the hybrid reproducing kernel. First, the hybrid reproducing kernel can only reflect global characteristics of graph structures, thus it is based on the entropies of global graph structures.

By contrast, the proposed kernel  $k_{\text{MELG}}$  is based on the nest depth-based representation that can simultaneously reflect the local and global entropy-based graph characteristics. Second, as we have stated, the largest layer expansion subgraph of a graph rooted at the centroid vertex is just the global structure of the graph, the hybrid reproducing kernel can be seen as the basic reproducing kernel between the largest layer expansion subgraphs. As a result, *the original hybrid reproducing kernel is just a special case of the proposed kernel*. Third, unlike the hybrid reproducing kernel, only the proposed kernel can accommodate label attributed graphs.

### 3.2 Computational Analysis

In this subsection, we analyze the computational complexity of the proposed mixed entropy local-global graph kernel. Assume we have a pair of graphs each having  $n$  vertices and  $m$  edges. Computing the family of expansion subgraphs for each graph relies on the computation of the shortest path matrix and requires time complexity  $O(m \log n)$ . Moreover, computing the nest depth-based representation of each graph through its expansion subgraphs relies on the computation of the entropy measure on each of the subgraphs, and requires time complexity  $O(Ln^2)$ . Here,  $L$  is the greatest length of the shortest paths rooted at the centroid vertices of all graphs and  $L \ll n$ . Finally, for  $k_{\text{MELG}}$ , computing the required reproducing kernel  $K_1$  between the entropies of  $L$  pairs of  $K$ -layer expansion subgraphs requires time complexity  $O(L)$ . As a result, the proposed kernel  $k_{\text{MELG}}$  has polynomial time complexity  $O(m \log n + L + Ln^2)$ . This computational analysis indicates that the new proposed kernel can be computed in a polynomial time.

## 4 Experimental Evaluations

In this subsection, we explore the performance of the proposed mixed entropy local-global kernel (MELG) on graph classification problems. Specifically, the standard graph datasets employed in the evaluation are the MUTAG, PTC, COIL5, Shock, CATH2, Reeb and D&D. Details of these datasets are shown in Table 1. Moreover, we compare the proposed MELG kernel with five state-of-the-art kernels, including the Jensen-Shannon graph kernel (JSGK) [3], the random walk graph kernel (RWGK) [12], the Lovász graph kernel (LGK) [11], the nested alignment local-global kernel (NALG) [5], and the hybrid reproducing kernel (HRK) [17]. The RWGK kernel is a typical example of local kernel that relies on local random walk substructures. The LGK, HRK and JSGK kernels are global kernels that can reflect the global characteristics of whole graph structures. The NALG kernel is a local-global graph kernel that can capture both the local and global graph characteristics. We compute the kernel matrix associated with each kernel on each dataset. We perform 10-fold cross-validation using a C-Support Vector Machine (C-SVM) to compute the classification accuracies, using LIBSVM software library [7]. We use nine samples for training and one for

**Table 1.** Information on the selected graph based bioinformatics datasets

Datasets	MUTAG	PTC	COIL5	Shock	CATH2	Reeb	D&D
Max # vertices	28	109	241	33	568	220	5748
Min # vertices	10	2	72	4	143	41	30
Mean # vertices	17.93	25.60	144.90	109.63	308.03	95.42	284.3
Max # edges	33	108	702	32	2220	219	14267
Min # edges	10	1	206	3	556	40	63
Mean # edges	19.79	25.96	419	12.16	1254.80	94.59	715.65
# graphs	188	344	360	150	190	300	1178
# classes	2	2	5	5	2	15	2
Mean# edges/Mean# vertices	1.10	1.00	2.89	0.92	4.07	0.99	2.52

testing. The parameters of the C-SVMs are optimized on each training set using cross-validation. We report the average classification accuracy ( $\pm$ standard error) and the runtime for each kernel in Tables 2 and 3. The runtime is measured under Matlab R2015a running on a 2.5 GHz Intel 2-Core processor (i.e., i5-3210m).

**Table 2.** Classification accuracy (in %  $\pm$  standard error) runtime in second.

Datasets	MUTAG	PTC	COIL5	Shock	CATH2	Reeb	D&D
MELG	<b>84.46</b> $\pm$ .50	57.28 $\pm$ .51	<b>71.41</b> $\pm$ .46	<b>40.06</b> $\pm$ .60	<b>74.14</b> $\pm$ .57	<b>45.63</b> $\pm$ .62	<b>75.81</b> $\pm$ .26
NALG	84.22 $\pm$ .50	58.00 $\pm$ .64	69.75 $\pm$ .65	37.60 $\pm$ .62	74.00 $\pm$ .83	45.20 $\pm$ .33	75.52 $\pm$ .31
JSGK	83.11 $\pm$ .80	57.29 $\pm$ .41	69.13 $\pm$ .79	21.73 $\pm$ .76	72.26 $\pm$ .76	21.73 $\pm$ .76	72.26 $\pm$ .76
RWGK	80.77 $\pm$ .75	53.97 $\pm$ .31	14.21 $\pm$ .65	0.33 $\pm$ .37	—	43.23 $\pm$ .30	—
LGK	80.83 $\pm$ .43	56.29 $\pm$ .47	—	31.80 $\pm$ .89	—	—	—
HRK	84.35 $\pm$ .51	58.23 $\pm$ .55	70.66 $\pm$ .49	37.93 $\pm$ .70	71.15 $\pm$ .68	27.40 $\pm$ .35	75.36 $\pm$ .54

In terms of the classification accuracies, it is clear that the proposed MELG kernel can outperform any alternative graph kernel on any dataset, excluding the HRK kernel on the D&D dataset. However, the proposed MELG kernel is still competitive to the HRK kernel on the D&D dataset. The reasons of the effectiveness are twofold. First, unlike the alternative JSGK, RWGK, LGK and HRK kernels that only reflect local or global graph characteristics, the proposed MELG kernel can simultaneously reflect both the local and global graph characteristics. Second, on the other hand, although the NALG kernel can also simultaneously reflect both the local and global graph characteristics. Only the proposed NALG kernel can accommodate vertex labels. In terms of the runtime, it is clear that the proposed MELG kernel has efficient computational complexity. By contrast, some alternative graph kernels cannot finish the computation on graph datasets with large graphs, e.g., a graph with thousands of vertices. In summary, the above experiments demonstrate the effectiveness and efficiency of the proposed kernel.

**Table 3.** Runtime for various kernels.

Datasets	MUTAG	PTC	COIL5	Shock	CATH2	Reeb	D& D
MELG	$1.0 \cdot 10^1$	$2.0 \cdot 10^0$	$8.0 \cdot 10^0$	$1.0 \cdot 10^0$	$1.0 \cdot 10^1$	$4.0 \cdot 10^0$	$1.5 \cdot 10^2$
NALG	$8.6 \cdot 10^2$	$2.3 \cdot 10^3$	$3.3 \cdot 10^3$	$3.8 \cdot 10^2$	$9.4 \cdot 10^2$	$1.3 \cdot 10^1$	$4.6 \cdot 10^2$
JSGK	$1.0 \cdot 10^0$	$1.0 \cdot 10^0$	$1.0 \cdot 10^0$	$1.0 \cdot 10^0$	$1.0 \cdot 10^0$	$1.0 \cdot 10^0$	$1.0 \cdot 10^0$
RWGK	$4.6 \cdot 10^1$	$6.7 \cdot 10^1$	$1.1 \cdot 10^3$	$2.3 \cdot 10^1$	–	$1.2 \cdot 10^3$	–
LGK	$1.0 \cdot 10^3$	$7.4 \cdot 10^3$	–	$1.0 \cdot 10^3$	–	–	–
HRK	$3.0 \cdot 10^0$	$1.3 \cdot 10^1$	$1.5 \cdot 10^1$	$2.0 \cdot 10^0$	$4.0 \cdot 10^0$	$9.0 \cdot 10^0$	$1.5 \cdot 10^2$

## 5 Conclusion

In this paper, we have proposed a new nest reproducing kernel for graphs. This kernel is based on a new reproducing kernel associated with the depth-based complexity traces. Since the computation of the complexity trace is only quadratic in the vertex number. Moreover, complexity trace of a graph is a nest sequence that can simultaneously encapsulate both the local and global entropy-based information. As a result, the proposed kernel can not only be efficiently computed but also simultaneously consider local and global graph characteristics of graph structural information. The experiments have demonstrated the effectiveness and efficiency of the proposed kernel.

**Acknowledgments.** This work is supported by the National Natural Science Foundation of China (Grant no. 61602535, 61503422 and 61773415), the Open Projects Program of National Laboratory of Pattern Recognition, and the program for innovation research in Central University of Finance and Economics.

## References

1. Alvarez, M.A., Qi, X., Yan, C.: A shortest-path graph kernel for estimating gene product semantic similarity. *J. Biomed. Semant.* **2**, 3 (2011)
2. Aronszajn, N.: Theory of reproducing kernels. *Trans. Am. Math. Soc.* **68**(3), 337–404 (1950)
3. Bai, L., Hancock, E.R.: Graph kernels from the Jensen-Shannon divergence. *J. Math. Imaging Vis.* **47**(1–2), 60–69 (2013)
4. Bai, L., Hancock, E.R.: Depth-based complexity traces of graphs. *Pattern Recogn.* **47**(3), 1172–1186 (2014)
5. Bai, L., Cui, L., Rossi, L., Xu, L., Hancock, E.R.: A nested alignment graph kernel through the dynamic time warping framework. *Pattern Recogn. Lett.* (to appear)
6. Bai, L., Rossi, L., Torsello, A., Hancock, E.R.: A quantum Jensen-Shannon graph kernel for unattributed graphs. *Pattern Recogn.* **48**(2), 344–355 (2015)
7. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2**(3), 27 (2011)
8. Costa, F., De Grave, K.: Fast neighborhood subgraph pairwise distance kernel. In: *Proceedings ICML*, pp. 255–262 (2010)

9. Dehmer, M., Mowshowitz, A.: A history of graph entropy measures. *Inf. Sci.* **181**(1), 57–78 (2011)
10. Han, L., Escolano, F., Hancock, E.R., Wilson, R.C.: Graph characterizations from von Neumann entropy. *Pattern Recogn. Lett.* **33**(15), 1958–1967 (2012)
11. Johansson, F., Jethava, V., Dubhashi, D., Bhattacharyya, C.: Global graph kernels using geometric embeddings. In: *Proceedings of ICML*, pp. 694–702 (2014)
12. Kashima, H., Tsuda, K., Inokuchi, A.: Marginalized kernels between labeled graphs. In: *Proceedings of ICML*, pp. 321–328 (2003)
13. Kriege, N., Mutzel, P.: Subgraph matching kernels for attributed graphs. In: *Proceedings of ICML* (2012)
14. Riesen, K., Bunke, H.: *Graph Classification and Clustering Based on Vector Space Embedding*. World Scientific Publishing Co., Inc., River Edge (2010)
15. Rossi, L., Torsello, A., Hancock, E.R., Wilson, R.C.: Characterizing graph symmetries through quantum Jensen-Shannon divergence. *Phys. Rev. E* **88**(3), 032806 (2013)
16. Urry, M., Sollich, P.: Random walk kernels and learning curves for Gaussian process regression on random graphs. *J. Mach. Learn. Res.* **14**(1), 1801–1835 (2013)
17. Xu, L., Jiang, X., Bai, L., Xiao, J., Luo, B.: A hybrid reproducing graph kernel based on information entropy. *Pattern Recogn.* **73**, 89–98 (2018)
18. Xu, L., Niu, X., Xie, J., Abel, A., Luo, B.: A local-global mixed kernel with reproducing property. *Neurocomputing* **168**, 190–199 (2015)
19. Xu, L., Chen, X., Niu, X., Zhang, C., Luo, B.: A multiple attributes convolution kernel with reproducing property. *Pattern Anal. Appl.* **20**(2), 485–494 (2017)



# Dirichlet Densifiers: Beyond Constraining the Spectral Gap

Manuel Curado<sup>1</sup>(✉), Francisco Escolano<sup>1</sup>, Miguel Angel Lozano<sup>1</sup>,  
and Edwin R. Hancock<sup>2</sup>

<sup>1</sup> University of Alicante, Alicante, Spain  
{mcurado,sco,malozano}@dccia.ua.es

<sup>2</sup> University of York, York, UK  
edwin.hancock@york.ac.uk

**Abstract.** In this paper, we derive a new bound for commute times estimation. This bound does not rely on the spectral gap but on graph densification (or graph rewiring). Firstly, we motivate the bound by showing that implicitly constraining the spectral gap through graph densification cannot fully explain some estimations in real datasets. Then, we set our working hypothesis: if densification can deal with a small/moderate degradation of the spectral gap, this is due to the fact that inter-cluster commute distances are considerably shrunk. This suggests a more detailed bound which explicitly accounts for the shrinking effect of densification. Finally, we formally develop this bound, thus uncovering the deep implications of graph densification in commute times estimation.

**Keywords:** Graph densification · Commute times  
Spectral graph theory

## 1 Introduction

Given an input graph  $G = (V, E)$ , *graph densification* produces a graph  $H = (V, E')$ , where  $E \subset E'$ . This concept was formalized by Hardt and coworkers [6] as a means of ruling out non-trivial graph embeddings. For instance, they proved that a graph can be densified if and only if cannot be embedded under a weak notion of embeddability. Originally, the purpose of this characterization is to understand structural differences between sparse graphs and dense graphs in order to reduce the complexity of several combinatorial problems: the MAX-CUT problem, which is NP-hard, has a PTAS (Polynomial Time Approximation Scheme) when its associated graph is dense [2].

More recently, graph densification has been considered as an interesting tool for structural pattern recognition. Escolano et al. [4, 5] have exploited the fact that densification often requires cut preservation, in order to conjecture that densified graphs can be better conditioned for spectral clustering than their un-densified counterparts. In this regard, it is well known that commute times suffer from the problem of *global information loss*. More precisely, von Luxburg



et al. [8] showed that commute times are diffused through the graph in such a way that the local part of the diffusion (in the neighborhood of both the origin and destination nodes) dominates the global one (inside the graph).

We conjecture that densification provides an effective way to provide more clustered subgraphs so that the commute times can be shrunk for inter-cluster nodes, thus providing a more effective estimation of these inter-node distances, so that they cannot be confused with larger inter-cluster distances. If so, we need tighter bounds for commute times estimations with respect to the usual bound relying on constraining the spectral gap.

In this paper, we review the Dirichlet densification algorithm, which typically doubles the number of edges with respect to the original graph. Then, we analyze the von Luxburg et al.'s bound, which relies on the spectral gap, and present its limitations in practice. This motivates a more detailed analysis that lead us to introduce a novel bound for commute times (scaled effective resistance) estimations.

## 2 Dirichlet Densifiers

The Dirichlet approach to densification [5] consists of the following steps:

1. **Knn-graph:** Given a data set  $\chi = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \subset \mathbb{R}^d$ , we map the  $\mathbf{x}_i$  to the vertices  $V$  of an undirected weighted graph  $G(V, E, W)$  with  $W_{ij} = e^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / \sigma^2}$  and  $(i, j) \in E$  if  $W_{ij} > 0$  and  $j \in N_k(i)$ .
2. **Return Random Walk:** Given  $G = (V, E, W)$  reformulate  $W$  in terms of  $W_e$  so that

$$W_{e_{ij}} = \max_k \max_{\forall l \neq k} \{p_{v_k}(v_j|v_i)p_{v_l}(v_i|v_j)\}, \tag{1}$$

where  $p_{v_k}(v_j|v_i) = \frac{W_{ik}W_{kj}}{d(v_i)d(v_j)}$ ,  $p_{v_l}(v_i|v_j) = \frac{W_{jl}W_{li}}{d(v_j)d(v_i)}$  (*go* and *return* probabilities, respectively) and  $d(\cdot)$  is the degree function. Therefore,  $W_{e_{ij}}$  relies on maximizing the probability that a random walk goes from  $i$  to  $j$  through  $l$  and then returns through a different vertex  $k$ . This strategy minimizes the weight of spurious inter-class links.

3. **Edge Selection:** Given  $G' = (V, E, W_e)$ , select  $E'' \subset E$ , with  $|E''| \ll |E|$  as follows:
  - (a)  $\mathcal{S} = \text{sort}(E, W_e, \text{descend})$ .
  - (b)  $\mathcal{S}' = \mathcal{S} \sim \{e \in \mathcal{S} : W_e < \delta_1\}$  where  $\delta_1$  is set so that  $|\mathcal{S}'| = \alpha|\mathcal{S}|$ .
4. **Line Graph:** Given  $G'' = (V, \mathcal{S}', W_e)$  construct a the graph

$$\text{Line} = (\mathcal{S}', \text{Line}_E, \text{Line}_{W_e})$$

where

- (a) The nodes of  $e_i \in \text{Line}$  are the edges in  $\mathcal{S}'$ .
- (b) The weight function  $\text{Line}_{W_e}$  is defined as follows:

$$\text{Line}_{W_e}(e_a, e_b) = \sum_{k=1}^{|E''|} p_{e_k}(e_b|e_a)p_{e_k}(e_a|e_b), \tag{2}$$

i.e. we use *go* and *return* probabilities.

(c)  $Line_E = \{(e_a, e_b) : Line_{W_e}(e_a, e_b) > 0\}$ .

5. **Dirichlet Process:** Given the *Line* graph, we proceed as follows:

(a)  $\mathcal{SB} = \text{sort}(\mathcal{S}', Line_W, \text{descend})$ .

(b)  $\mathcal{SB}' = \mathcal{SB} \sim \{e \in Line_E : Line_{W_e} < \delta_2\}$  where  $\delta_2$  is set so that  $|\mathcal{SB}'| = \beta|\mathcal{SB}|$ .

(c) Consider  $\mathcal{SB}'$  as the boundary  $B$  (known labels) of a Dirichlet process driven by the Laplacian  $Line_{\mathcal{L}} = Line_D - Line_{W_e}$ . Then, finding an harmonic function, i.e. a function  $u(\cdot)$  satisfying  $\nabla^2 u = 0$  consists of minimizing:

$$D_{Line}[u] = \frac{1}{2}u^T Line_{\mathcal{L}}u \tag{3}$$

where  $u = [u_B, u_I]$  and  $Line_{\mathcal{L}}$  are re-ordered so that the boundary nodes (edges in *Line*) come first. Then, minimizing  $D_{Line}[u]$  w.r.t.  $u_I$  leads to label of the unknown nodes (edges in *Line*)  $u_I$  as the solutions to the following linear system:

$$L_I u_I = -K^T u_B, \tag{4}$$

where all the  $u_B$  are set to the unit,  $L_I$  is the sub-Laplacian of  $Line_{\mathcal{L}}$  concerning the  $u_I$  nodes, and  $K$  is a  $|\mathcal{SB}'| \times |\mathcal{SB}'|$  block of the re-ordered Laplacian.

6. **Relabelling:** Since there is a bijection between the nodes in the line graph and the edges in the original graph, we relabel the edges in the original graph with the information coming from the Dirichlet process in the line graph.

**Table 1.** NIST: Adjusted Rand Index for different thresholds and number of  $k$

		kNN 15			kNN 25			kNN 35		
		E <sub>B</sub>								
		0.05	0.25	0.5	0.05	0.25	0.5	0.05	0.25	0.5
E''	0.05	37.3	41.88	40.62	57.23	54.33	52.26	27.12	30.88	43.49
	0.15	66.9	63.52	61.64	70.87	70.84	57.65	69.51	68.54	67.42
	0.25	71.78	69.15	65.01	71.05	70.4	70.21	69.95	71.6	70.51
	0.35	<b>74.4</b>	71.06	70.08	71.02	71.51	70.42	70.55	71.23	70.49
No dense		69.25			65.62			63.74		

### 3 A Novel Densification-Based Bound

#### 3.1 The von Luxburg et al. Bound

The starting point of our approach is the following bound, derived by von Luxburg et al. [8] for any connected, undirected graph  $G = (V, E)$  that is not bipartite:

$$\left| \frac{1}{vol(G)} CT_{st} - \left( \frac{1}{d_s} + \frac{1}{d_t} \right) \right| \leq 2 \left( \frac{1}{\lambda_2} + 2 \right) \frac{w_{max}}{d_{min}^2} \tag{5}$$

**Table 2.** NIST: spectral gaps for different thresholds and number of  $k$ 

		kNN 15			kNN 25			kNN 35		
		$ E_B $								
		0.05	0.25	0.5	0.05	0.25	0.5	0.05	0.25	0.5
$ E'' $	0.05	0.0	0.0	0.0	0.0209	0.0251	1.9561	0.0498	0.0478	0.0395
	0.15	0.0049	0.0	0.0	0.0310	0.0275	0.0233	0.0778	0.0714	0.0630
	0.25	0.0097	0.0	0.0	0.0446	0.0356	0.0290	0.1043	0.0899	0.0732
	0.35	<b>0.0176</b>	0.0130	0.0073	0.0632	0.0478	0.0323	0.1337	0.1120	0.0865
No dense		0.0192			0.0481			0.0775		

where  $CT_{st} = R_{st} \text{vol}(G)$  is the commute time between the nodes  $s$  and  $t$ ,  $R_{st}$  is the effective resistance,  $\text{vol}(G)$  is the volume of the graph,  $\lambda_2$  is the *spectral gap* and  $d_{\min}$  is the minimum node degree in  $G$ . The spectral gap  $\lambda_2$  is the second eigenvalue of the normalized graph Laplacian  $L = I - D^{-1}W$  where  $D = \text{diag}(d_1, \dots, d_n)$  is the degree matrix and  $W$  is the (symmetric) affinity matrix, with  $w_{ij} > 0$  if  $(i, j) \in E$ . Then  $w_{\max}$  is the maximal affinity.

The above equation explains why commute times are meaningless in large graphs. These graphs tend to have large spectral gaps due to the existence of inter-cluster links (noise). As a result, we have  $R_{st} \approx \frac{1}{d_s} + \frac{1}{d_t}$ , i.e. commute times do only depend on their local degrees. Consequently they are meaningless for measuring distances between nodes in large graphs.

Conversely, a way of making  $R_{st} \approx \frac{1}{d_s} + \frac{1}{d_t}$  diverge (and thus make commute times meaningful) is to reweight/rewire the edges in  $E$  so that  $\lambda_2 \rightarrow 0$ . This task is *partially due* by graph densification, which implicitly constrains the spectral gap as much as possible. Our preliminary experiments show that Dirichlet densifiers (algorithm described in Sect. 2) lead to improve the Adjusted Rand Index (ARI) obtained from commute times *after densification* in a variety of datasets (NIST<sup>1</sup>, COIL-20<sup>2</sup>, FlickrLOGOs-32<sup>3</sup> and YALE-Faces<sup>4</sup>).

To motivate our discussion, in Table 1 we show the ARIs obtained for the NIST dataset in several scenarios. Each scenario is characterized by: (1) a value  $k$  for building the  $k$ -NN, (2) the fraction  $|E''|$  of dominating edges chosen for building the line graph, and (3) the fraction of dominating  $|E_B|$  edges chosen as seeds for the harmonic analysis (Dirichlet process). In all scenarios, the ARIs *before densifying the datasets* is below 70% (decreases as  $k$  increases). The question addressed by densification is whether this performance can be improved by rewiring/densifying the similarity graphs. Our analysis shows that for a small fraction of  $|E''|$  (typically 0.35) and a tiny fraction of  $|E_B|$  (typically 0.05) densification significantly improves the commute times of the input graphs (best result ARI=74.4%).

<sup>1</sup> <http://yann.lecun.com/exdb/mnist/>.

<sup>2</sup> <http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php>.

<sup>3</sup> <http://www.multimedia-computing.de/flickrlogos/>.

<sup>4</sup> <http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html>.

A detailed interpretation of the above ARIs leads to evaluate the bound in Eq. 5 from the perspective of the spectral gap  $\lambda_2$ . In other words, we want to quantify the real effect of constraining the spectral gap in the improvement of the commute times estimation. In Table 2, we show the spectral gaps for all the scenarios. In general, the larger the spectral gap the worse the performance, as expected. We remove from the analysis the disconnected graphs ( $\lambda_2 = 0$ ) arising in the scenarios for  $k = 15$  since they are not contemplated by the bound. However, as  $k$  increases ( $k = 25, k = 35$ ), we find some contradictions. For densified graphs, we have that larger gaps than those of the respective not-densified graphs outperform their ARIs in some cases (specially for optimal configurations).

The above results suggest that the von Luxburg et al.’s bound (Eq. 5) does not fully characterize the effect of densification. Our working hypothesis is that constraining the spectral gap is only *part of the process* of re-estimating commute times for large graphs. Of course, the spectral gap has to be kept as smaller as possible for a reliable estimation of commute times. However, this becomes more and more difficult as  $k$  grows due to the appearance of inter-cluster links. Thus, if densification can deal with a small/moderate degradation of the spectral gap, this is due to the fact that inter-cluster commute distances are considerably shrunk. This suggests a more detailed bound which explicitly accounts for the effect of densification.

### 3.2 The Proposed Bound

Given a graph  $G = (V, E)$  and two nodes  $s, t \in V$ , the commute time  $CT_{ij}$  is the expected time it takes a random walk to travel from  $s$  to  $t$  and back [3, 7, 9]. The link  $R_{st} = \frac{1}{vol(G)}CT_{ij}$ , where  $R_{st}$  is the effective resistance, characterizes the diffusive nature of CTs:

$$R_{st} \triangleq \arg \min_Y \sum_{e \in E} r_e |y_e|^p,$$

with  $p = 2$ , and  $Y \triangleq \{y_e\}_{e \in E}$  is the unit flow from  $s$  to  $t$  (inject a unit current at  $s$ , extract it at  $t$  and observe the flow traced across the edges  $e \in E$ ). Unit flows have two interesting properties: (1) they are quite scattered along the edges (even in moderate size graphs), and (2) the bulk of their magnitude is in the neighborhood of both  $s$  and  $t$ .

Effective resistances also satisfy the Rayleigh monotonicity principle: Given  $G$  with adjacency/similarity matrix  $W$ , let  $G'$  with adjacency/similarity  $W'$  which is identical to  $W$  except for the increase in the weight of one arbitrary edge  $(i, j)$ , so that  $W'_{ij} = W_{ij} + \delta$ . Then, for arbitrary vertices  $s$  and  $t$ , we have

$$R^G(s, t) \geq R^{G'}(s, t),$$

i.e. introducing new edges (or reweighting them incrementally) does not increase the effective resistance between any pair of nodes  $s$  and  $t$  in the graph. Thus, in order to quantify the effect of densification in bounding the effective resistance, we will exploit this principle as follows.

### 3.3 Upper Bound

Let  $G = (V, E)$  be an undirected and unweighted graph ( $r_e = 1$  for  $e \in E$ ), with  $n = |V|$  and average degree  $\tau = \Theta(d)$ . Given any pair of nodes,  $s$  and  $t$ , let  $Y \triangleq \{y_e\}_{e \in E}$  be any unit flow between these nodes, and  $Y^* \triangleq \{y_e^*\}_{e \in E}$  the minimal flow that defines the effective resistance  $R^G(s, t) = \sum_{e \in E} |y_e^*|^2$ . As a result:  $R^G(s, t) \leq \sum_{e \in E} |y_e|^2$ . Consequently, we will construct a tight upper bound for  $R^G(s, t)$  (as in [1]) and then we will show that when  $G$  is densified, leading to  $H = (V, E')$  with  $E \subset E'$ , the bound associated with  $R^H(s, t)$  is even tighter.

The flow  $Y \triangleq \{y_e\}_{e \in E}$  is constructed as follows:

- (1) Start at  $s$  by injecting a unit flow. The local flow sent to any of the  $N_1$  neighbours of  $s$  is  $1/d_s$ . Their contribution to  $Y$  is  $1/d_s$ .
- (2) The flow must be *unitary* (input flow equal to output flow for each node, until arriving to destination  $t$ ). Thus, any of the  $N_2$  neighbours of  $N_1$  must diffuse a flow  $1/(N_2 d_s)$ . Then, let  $S$  be the number of *layers* with successive neighbours  $N_1, N_2, \dots, N_S$ . Since  $N_k = \tau k$ , we have that, if any neighbour diffuses  $1/N_k$  then

$$R^G(s, t) \leq \frac{1}{d_s} + \frac{1}{\tau^2} \sum_{k=1}^S \frac{1}{k}. \tag{6}$$

The value of  $S$  depends on the graph and it is not constant but for balanced trees (see Fig. 1). Thus, the bound in Eq. 6 is an upper bound derived from setting  $S$  as the maximum reachable neighbourhood according to unitary diffusion. This means that there exists a symmetric process starting from the destination node  $t$ . W.o.l.g. (for the definition of a bound) we can assume that this symmetric process has also  $S$  layers. Then:

$$R^G(s, t) \leq \frac{1}{d_s} + \frac{1}{d_t} + 2 \frac{1}{\tau^2} \sum_{k=1}^S \frac{1}{k}. \tag{7}$$

- (3) Finally, to have a unit flow, we must link the two last layers (the one coming from  $s$  and that coming from  $t$ ) through some of the existing edges between the nodes of these final layers so that only a flow of  $1/N_S$  per node is transferred in order to ensure unitarity. Then:

$$R^G(s, t) \leq \frac{1}{d_s^2} + \frac{1}{d_t^2} + 2 \frac{1}{\tau^2} \sum_{k=1}^S \frac{1}{k} + \frac{1}{\tau^2} \cdot \frac{1}{S}. \tag{8}$$

What happens after densification? We can summarize it as redefining  $\tau$  (the average degree) in terms of  $q\tau$ . In particular, Dirichlet densifiers assume  $q = 2$  (two transitive edges are linked by an additional one). Then, for a densified graph  $H$  using a Dirichlet process, the bound in 8 is redefined as

$$R^H(s, t) \leq \frac{1}{d_s^2} + \frac{1}{d_t^2} + \frac{1}{2\tau^2} \sum_{k=1}^S \frac{1}{k} + \frac{1}{4\tau^2} \cdot \frac{1}{S}, \tag{9}$$

which reduces the bound for  $G$  in at least  $1/4$  of the flow propagated through the  $S$  layers in one sense (either from  $s$  to  $t$  or vice versa).

### 3.4 Lower Bound

For the lower bound of  $R^G(s, t)$  one must consider that the Rayleigh principle allows to construct a graph  $G'$  as follows (see also [1, 3]).  $G'$  is a linear contracted graph following the line connecting  $s$  and  $t$ . We start with node  $s$  and add edges of resistance 0 between all the neighbours of  $s$  and merge all these nodes in a single node  $v_1$ . These edges form a *slice*. We repeat this process for nodes  $v_2, \dots, v_S$  where  $E_j$  are the edges associated with the slice between  $v_j$  and  $v_{j+1}$ . Finally we add a final slice between  $v_S$  and  $t$ . This construction is useful because: (1) it is ideal for a lower bound because suppressing edges in the original graph increases the effective resistance, (2) the flow between  $v_j$  and  $v_{j+1}$  is always unitary, and (3) the edges  $E_j$  lead to an inverse parallel resistance according to the law  $1/r = 1/r_1 + 1/r_2$ . More precisely:

$$R^{G'}(s, t) = \sum_{e \in E} i_e^2 = \frac{1}{d_s} + \sum_{j=0}^S \sum_{k=1}^{E_j} i_k^2 + \frac{1}{d_t}. \tag{10}$$

According to the generalized mean inequality we have:

$$\sum_{k=1}^{E_j} i_k^2 \geq \sum_{j=0}^S \frac{1}{E_j} \underbrace{\left( \sum_{k=1}^{E_j} i_k \right)}_1 = \sum_{j=0}^S \frac{1}{E_j}.$$

Therefore, since  $G'$  has less edges than  $G$ , then  $R^{G'}(s, t) \geq R^G(s, t)$  we have the following bound for un-densified graph:

$$R^G(s, t) \geq \frac{1}{d_s} + \frac{1}{d_t} + \sum_{j=0}^S \frac{1}{E_j} \geq \frac{1}{d_s} + \frac{1}{d_t} + \frac{S-1}{E_{max}}, \tag{11}$$

where  $E_{max}$  the maximal number of edges in a slice.

Now, If we densify  $G$  leading to  $H = (V, E')$  with  $E \subset E'$ , we have that all the slices between  $v_j$  and  $v_{j+1}$  must have more edges  $E'_j \geq E_j$ . This is due to the fact that few of them must be zeroed in comparison to those retained to form slices. This leads to  $E'_{max} \geq E_{max}$ , where  $E'_{max}$  is the maximal number of edges in a slice for the contracted  $H$ . As a result a smaller lower bound (i.e. effective resistance can be significantly lower in a densified graph).

$$R^H(s, t) \geq \frac{1}{d_s} + \frac{1}{d_t} + \frac{S-1}{E'_{max}} \tag{12}$$

In addition, it is more difficult to create the contracted graph in a densified graph since there are links between different slices. Such links contribute to reduce the minimal effective resistance even more. However, we can assume that the bulk of the contribution to the effective resistance is on the removed edges, i.e. in the process of retaining  $E'_j > E_j$  in each slice<sup>5</sup>.

Concerning densification, it is important to set the loss of  $E'_{max}$  with respect to  $E_{max}$ . For Dirichlet densifiers, we can assume  $E'_{max} = 2E_{max}$ .

### 3.5 The Proposed Bound

As a result, we have that for a densified graph  $H$  we have the following bounds for any effective resistance:

$$R_{approx} + \frac{1}{2} \cdot \frac{S-1}{E_{max}} \leq R^H(s, t) \leq R_{approx} + \frac{1}{2} \cdot \frac{1}{\tau^2} \sum_{k=1}^S \frac{1}{k} + \frac{1}{4\tau^2} \cdot \frac{1}{S}, \quad (13)$$

where  $R_{approx} = 1/d_s + 1/d_t$ . With respect to the same bound for the not-densified graph  $G$ :

$$R_{approx} + \frac{S-1}{E_{max}} \leq R^G(s, t) \leq R_{approx} + 2 \frac{1}{\tau^2} \sum_{k=1}^S \frac{1}{k} + \frac{1}{\tau^2} \cdot \frac{1}{S}, \quad (14)$$

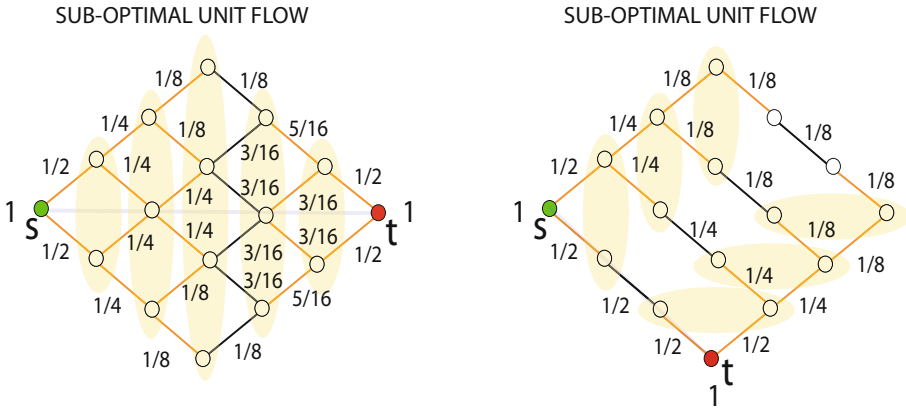
Summarizing, densification reduces significantly (1/2) the upper bound and also reduces (1/4) the upper bound associated with not-densified graphs. This is because  $q = 2$  for Dirichlet densifiers.

## 4 Discussion and Conclusion

In this paper, we have analyzed the impact of graph densification in bounding effective resistances (scaled commute times). In this regard, we contribute with a novel bound, which is more detailed than that relying on the spectral gap  $\lambda_2$ . Although the spectral gap is linked with the density of the graph (it is upper bounded by the Cheeger constant), the analysis based on  $\lambda_2$  does only address the ratio between the smallest cut and graph density. However, the reformulation of the von Luxburg et al.'s bound requires to estimate the impact of densification in shrinking the inter-cluster commute distances, thus leading to better estimates than those provided for the original graph.

With the new bound at hand, we show that Dirichlet densification reduces significantly (1/2) the upper bound and also reduces (1/4) the upper bound associated with not-densified graphs. Simultaneously, since the Dirichlet procedure minimizes inter-cluster links, we have that the shrinkage in terms of commute

<sup>5</sup> Conversely, if this is not the case, we are forced to fuse more nodes, thus reducing the number of slices from  $S$  to, say,  $S'$  with more edges each. This leads to contracting the bound for  $R^H(s, t)$  even more.



**Fig. 1.** Examples of sub-optimal unit flows for bounding. Left: unit flow between  $s$  and  $t$  with the layers (upper bound) in yellow. Inter-layer links are in black. In this example there are  $S = 3 + 2$  layers. However if we change the destination node, then we have  $S = 3 + 3$  smaller layers. Since we have an upper bound we do not need to exploit all the edges in the graph to find the unit flow. (Color figure online)

distances is confined to intra-cluster nodes, thus leading to best ARIs (Adjusted Rand Indices) after commute times are estimated in densified graphs.

**Acknowledgments.** M. Curado, F. Escolano and M.A. Lozano are funded by the project TIN2015-69077-P of the Spanish Government.

**References**

1. Alamgir, M., von Luxburg, U.: Phase transition in the family of p-resistances. In: 25th Annual Conference on Neural Information Processing Systems. Advances in Neural Information Processing Systems. Proceedings of a Meeting Held at Granada, Spain, 12–14 December 2011, vol. 24, pp. 379–387 (2011)
2. Aroraa, S., Kargerb, D., Karpinskic, M.: Polynomial time approximation schemes for dense instances of NP-hard problems. *J. Comput. Syst. Sci.* **58**(1), 193–210 (1999)
3. Doyle, P.G., Snell, J.L.: *Random Walks and Electric Networks*, vol. 22, 1st edn. Mathematical Association of America, Washington, D.C. (1984)
4. Escolano, F., Curado, M., Hancock, E.R.: Commute times in dense graphs. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) *S+SSPR 2016*. LNCS, vol. 10029, pp. 241–251. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_22](https://doi.org/10.1007/978-3-319-49055-7_22)
5. Escolano, F., Curado, M., Lozano, M.A., Hancock, E.R.: Dirichlet graph densifiers. In: Robles-Kelly, A., Loog, M., Biggio, B., Escolano, F., Wilson, R. (eds.) *S+SSPR 2016*. LNCS, vol. 10029, pp. 185–195. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49055-7\\_17](https://doi.org/10.1007/978-3-319-49055-7_17)
6. Hardt, M., Srivastava, N., Tulsiani, M.: Graph densification. In: *Innovations in Theoretical Computer Science*, Cambridge, MA, USA, 8–10 January 2012, pp. 380–392 (2012)



7. Lovász, L.: Random walks on graphs: a survey. In: Miklós, D., Sós, V.T., Szőnyi, T. (eds.) *Combinatorics, Paul Erdős is Eighty*, vol. 2, pp. 353–398. János Bolyai Mathematical Society, Budapest (1996)
8. von Luxburg, U., Radl, A., Hein, M.: Hitting and commute times in large random neighborhood graphs. *J. Mach. Learn. Res.* **15**(1), 1751–1798 (2014)
9. Qiu, H., Hancock, E.R.: Clustering and embedding using commute times. *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(11), 1873–1890 (2007)

## Author Index

- Alberti, Michele 470  
Algul, Enes 439  
Al-Khafaji, Suhad Lateef 416  
Álvarez, José 86
- Bai, Lu 227, 237, 406, 491, 501  
Bai, Xiao 42, 107, 204, 386, 395  
Bernard, Simon 32  
Bicego, Manuele 119  
Blumenthal, David B. 293  
Boria, Nicolas 460  
Bottarelli, Lorenzo 160  
Bougleux, Sébastien 97, 293, 460  
Bouzaïeni, Abdessalem 3  
Brun, Luc 97, 293, 460
- Caglar, Ibrahim 217  
Cao, Hongliu 32  
Cardot, Hubert 326, 429  
Carletti, Vincenzo 315  
Chen, Guangliang 52  
Conte, Donatello 304, 326, 429  
Cortés, Xavier 326, 429  
Cui, Lixin 227, 237, 406, 491, 501  
Curado, Manuel 512
- da S. Torres, Ricardo 345  
Daller, Évariste 97  
Darwiche, Mostafa 304  
de O. Werneck, Rafael 345  
Deng, Xiaogang 76  
Dwivedi, Shri Prakash 337
- Escolano, Francisco 512
- Fischer, Andreas 470  
Foggia, Pasquale 315
- Gabdulkhakova, Aysylu 258  
Gamper, Johann 293  
Gao, Yaozong 14  
Gao, Yongsheng 248  
Greco, Antonio 315  
Guan, Shaoya 376
- Haindl, Michal 22  
Han, Lirong 76  
Hancock, Edwin R. 217, 237, 367, 449, 491, 501, 512  
Hancock, Edwin 42, 395  
Heutte, Laurent 32  
Hong, Haiyun 491  
Hu, Yiqun 406
- Ingold, Rolf 470
- Jiao, Yuhang 227, 237
- Kropatsch, Walter G. 258  
Krzyżak, Adam 194
- Langer, Bernhard W. 258  
Langer, Maximilian 258  
Lézoray, Olivier 97  
Li, Chenglong 150  
Li, Yue 248  
Liew, Alan Wee-Chung 416  
Liu, Jianming 248  
Liu, Yun 204, 386  
Liwicki, Marcus 470  
Loog, Marco 119, 160  
Lovato, Pietro 119  
Lozano, Miguel Angel 512  
Luo, Zhiheng 357
- Maergner, Paul 470  
Meng, Cai 376  
Mensi, Antonelli 119  
Mi, Jian-Xun 357  
Moreno-García, Carlos Francisco 271  
Motobayashi, Masahiro 184
- Odate, Ryosuke 184
- Pelillo, Marcello 481  
Pondenkandath, Vinaychandran 470
- Raab, Christoph 173  
Raveaux, Romain 304, 345

- Rayar, Frédéric 65, 140  
Rekik, Islem 14  
Remeš, Václav 22  
Ren, Peng 76  
Riesen, Kaspar 470  
Robles-Kelly, Antonio 86  
Rossi, Luca 237, 501
- Sabourin, Robert 32  
Saggese, Alessia 315  
Sandi, Giulia 481  
Santacruz, Pep 282  
Schleif, Frank-Michael 173  
Serratosa, Francesc 271, 282, 326  
Shen, Dinggang 14  
Shinjo, Hiroshi 184  
Singh, Ravi Shankar 337  
Suliman, Karima Ben 194  
Sun, Peng 107  
Suzuki, Yasufumi 184
- T\*Kindt, Vincent 304  
Tabbone, Salvatore 3, 345  
Tang, Jin 150  
Tang, Wenzhong 107  
Tax, David M. J. 119  
Tino, Peter 173
- Uchida, Seiichi 65, 140
- Valev, Ventzeslav 194  
Vascon, Sebastiano 481  
Vento, Mario 315
- Wang, Chen 204  
Wang, Jianjia 449  
Wang, Qi 376  
Wang, Qian 14  
Wang, Shuai 42  
Wang, Xiang 204  
Wang, Xinran 76  
Wang, Yue 227  
Wei, Ran 86  
Wilson, Richard C. 367, 439, 449  
Wu, Meihong 491
- Xie, Yi 376  
Xiong, Ziwei 150  
Xu, Lixiang 501  
Xu, Zhuobin 406, 491
- Yan, Cheng 386  
Yanev, Nicola 194  
Ye, Zhiling 406  
Yu, Leijian 76
- Zeng, Yangbin 491  
Zhang, Han 14  
Zhang, Lichi 14  
Zhang, Xueni 395  
Zhang, Zhihong 237, 406, 491, 501  
Zhao, Nan 150  
Zhou, Jun 42, 204, 386, 416  
Zhou, Lei 42, 395  
Zhu, Quanwei 357  
Zong, Xin 130