Unsupervised and Semi-Supervised Learning
Series Editor: M. Emre Celebi

Deepak P
Anna Jurek-Loughrey *Editors*

Linking and Mining Heterogeneous and Multi-view Data

Springer

Unsupervised and Semi-Supervised Learning

Springer's Unsupervised and Semi-Supervised Learning book series covers the latest theoretical and practical developments in unsupervised and semi-supervised learning. Titles – including monographs, contributed works, professional books, and textbooks – tackle various issues surrounding the proliferation of massive amounts of unlabeled data in many application domains and how unsupervised learning algorithms can automatically discover interesting and useful patterns in such data. The books discuss how these algorithms have found numerous applications including pattern recognition, market basket analysis, web mining, social network analysis, information retrieval, recommender systems, market research, intrusion detection, and fraud detection. Books also discuss semi-supervised algorithms, which can make use of both labeled and unlabeled data and can be useful in application domains where unlabeled data is abundant, yet it is possible to obtain a small amount of labeled data.

More information about this series at http://www.springer.com/series/15892

Deepak P • Anna Jurek-Loughrey
Editors

Linking and Mining Heterogeneous and Multi-view Data

Springer

Editors
Deepak P
Queen's University Belfast
Northern Ireland, UK

Anna Jurek-Loughrey
Queen's University Belfast
Northern Ireland, UK

ISSN 2522-848X ISSN 2522-8498 (electronic)
Unsupervised and Semi-Supervised Learning
ISBN 978-3-030-01871-9 ISBN 978-3-030-01872-6 (eBook)
https://doi.org/10.1007/978-3-030-01872-6

Library of Congress Control Number: 2018962409

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

Preface

With data being recorded at never seen before scales within an increasingly complex world, computing systems routinely need to deal with a large variety of data types from a plurality of data sources. More often than not, the same entity may manifest across different data sources differently. To be able to arrive at a well-rounded view across all data sources to fuel data-driven applications, such different entity representations may need to be linked, and such linked information be mined while being mindful of the differences between the different views. The goal of this volume is to offer a snapshot of the advances in this emerging area of data science and big data systems. The intended audience includes researchers and practitioners who work on applications and methodologies related to linking and mining heterogeneous and multi-view data.

The book covers a number of chapters on methods and algorithms—including both novel methods and surveys—and some that are more application-oriented and targeted towards specific domains. To keep the narrative interesting and to avoid monotony for the reader, chapters on methods and applications have been interleaved.

The first four chapters provide reviews of some of the foundational streams of research in the area. The first chapter is an overview of state-of-the-art methods for multi-view data completion, a task of much importance since each entity may not be represented fully across all different views. The second chapter surveys the field of multi-view data clustering, the most explored area of unsupervised learning within the realm of multi-view data. We then turn our attention to linking, with the third chapter taking stock of unsupervised and semi-supervised methods to linking data records from across data sources. Record linkage is a computationally expensive task, and blocking methods often come in handy to rein in the search space. Unsupervised and semi-supervised blocking techniques form the focus of the fourth chapter.

The fifth chapter is the first of application articles in the volume, and it surveys a rich field of application of multi-view data analytics, that of transportation systems that encompass a wide variety of sensors capturing different kinds of transportation data. Social media discussions often manifest with intricate structure along with the

textual content, and identifying discourse acts within such data forms the topic of the sixth chapter. The seventh chapter looks at imbalanced datasets with an eye on improving per-class classification accuracy. They propose novel methods that model the co-operation between different data views in order to improve classification accuracy. The eighth chapter is focused on an enterprise information retrieval scenario, where a fast and simple model for linking entities from domain-specific knowledge graphs to IR-style queries is proposed.

Chapter 9 takes us back to multi-view clustering and focuses on surveying methods that build upon non-negative matrix factorization and manifold learning for the task. The tenth chapter considers how heterogeneous data sources are leveraged in data science methods for fake news detection, an emerging area of high interest from a societal perspective. Chapter 11 is an expanded version of a conference paper that appeared in AISTATS 2018 and addresses the task of multi-view metric learning. Social media analytics, in addition to leveraging conventional types of data, is characterized by abundant usage of structure. Community detection is among the fundamental tasks within social media analytics; the last chapter in this volume considers evaluation of community detection methods for social media.

We hope that this volume, focused on both linking heterogeneous data sources and methods for analytics over linked data sources, will demonstrate the significant progress that has occurred in this field in recent years. We also hope that the developments reported in this volume will motivate further research in this exciting field.

Belfast, UK — Deepak P
Belfast, UK — Anna Jurek-Loughrey

Contents

Chapter 1
Multi-View Data Completion

Sahely Bhadra

Abstract Multi-view learning has been explored in various applications such as bioinformatics, natural language processing and multimedia analysis. Often multi-view learning methods commonly assume that full feature matrices or kernel matrices for all views are available. However, in partial data analytics, it is common that information from some sources is not available or missing for some data-points. Such lack of information can be categorized into two types. (1) *Incomplete view*: information of a data-point is partially missing in some views. (2) *Missing view*: information of a data-point is entirely missing in some views, but information for that data-point is fully available in other views (no partially missing data-point in a view).

Although multi-view learning in the presence of missing data has drawn a great amount of attention in the recent past and there are quite a lot of research papers on multi-view data completion, but there is no comprehensive introduction and review of current approaches on multi-view data completion. We address this gap in this chapter through describing the multi-view data completion methods.

In this chapter, we will mainly discuss existing methods to deal with *missing view* problem. We describe a simple taxonomy of the current approaches. And for each category, representative as well as newly proposed models are presented. We also attempt to identify promising avenues and point out some specific challenges which can hopefully promote further research in this rapidly developing field.

1.1 Introduction

Multi-View Data Multi-view learning is an emerging field to deal with data collected from multiple sources or "views" to utilize the complementary information in them. In case of multi-view learning, the scientific data are collected from diverse

S. Bhadra (✉)
Indian Institute of Technology, Palakkad, India
e-mail: sahely@iitpkd.ac.in; https://sites.google.com/iitpkd.ac.in/sahelybhadra/home

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_1

domains or obtained from various feature extraction techniques. For example, in current biomedical research, a large amount of data is collected from a single patient, such as clinical records (e.g., age, sex, histories, pathology and therapeutics) and high-throughput omics data (e.g., genomics, transcriptomics, proteomics and metabolomics measurements) [22]. Similarly in computer vision, multimedia data-set can simultaneously contain recorded audio as well as video that can come from different sources or cameras placed in different positions [24, 25, 38]. Also in case of natural language processing, data can be collected as content of the documents, meta information of the documents (e.g., title, author and journal) and possibly the co-citation network graph for scientific document management [11, 33].

The feature variables collected from multiple sources for each data-point therefore exhibit heterogeneous properties and hence can be naturally partitioned into groups. Each group is referred as a particular view. Each view contains one type of information which is possibly arriving from a single source and is more homogeneous. Figure 1.1 shows a pictorial presentation of multi-view data captured through multiple sources.

Multi-View Learning Existing algorithms for multi-view learning can be classified into three groups based on their main principle: (1) co-training, (2) subspace learning and (3) multiple kernel learning. Co-training [7, 21] is one of the earliest schemes for multi-view learning. The main aim of co-training is to maximize the mutual agreement between two distinct views of the data. The success of co-training lies in the assumption of data sufficiency in each view and information consistency among all views [39]. Whereas subspace learning-based approaches such as canonical correlation analysis (CCA) [15, 19] and factor analysis (FA) [8] aim to obtain a latent subspace shared by multiple views. Here the assumption is that, data in different views are generated from a global latent subspace along with some view-specific local factors. The task of learning the common latent subspace is based on observed data in all views. Thus, these methods need complete data availability in each view. For non-linear modeling of multi-view data the most popular method is the multiple kernel learning (MKL) [36], which was originally developed to control the search space capacity of possible kernel matrices to achieve good generalization [17]. Currently, MKL has been widely applied to problems involving multi-view data [10, 21, 37]. MKL defines multiple kernels corresponding to different views and combines them either linearly or non-linearly. However, for defining a kernel it needs complete data availability in each view.

Although existing multi-view learning algorithms are effective and have shown promising performance in different applications, they share a serious limitation. The success of most of the existing multi-view learning algorithms mainly relies on the assumption of completeness of data-sets, i.e., any multi-view example must always be fully observed in all views.

Missing Values Due to various failures or faults in collecting and pre-processing the data in different views, existing models are more likely to be challenged with missing information. Missing data-points, whether be partially or fully missing in

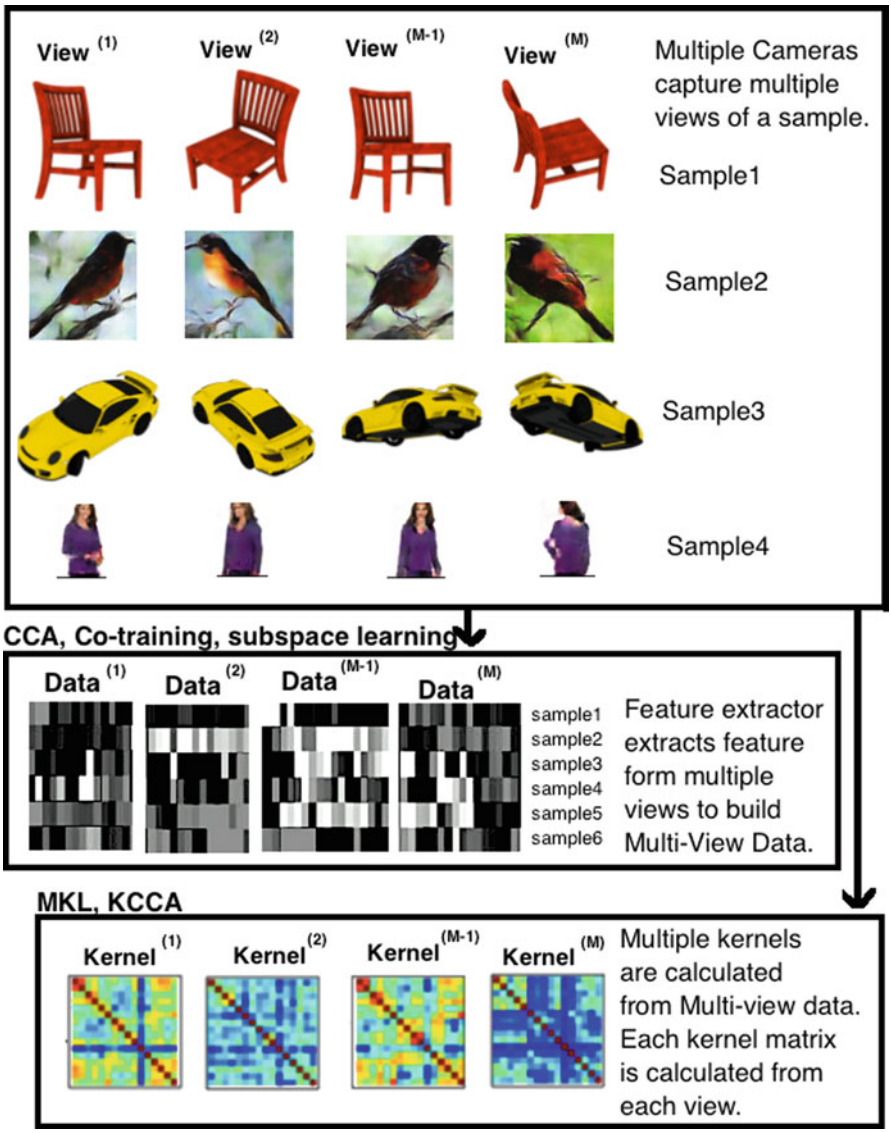


Fig. 1.1 Multi-view data when data about one data-point is captured by M different cameras from different positions resorting to different views

a view, exists in a wide range of fields such as social sciences, computer vision, biological systems and remote sensing (as in Fig. 1.2).

Missing View Often, information for a data-point corresponding to a view can be fully missing. As shown in Fig. 1.2 (top), for each data-point, the full picture from a view is totally missing for data-points while other pictures corresponding

to other views of that data-point are fully available. This kind of missing data-points will be denoted henceforth as data-point with *missing view*. For example, in remote sensing, some sensors can go off for some time, leaving gaps in the data. A second example is that when integrating legacy data-sets, some views may not be available for some data-points, because integration needs were not considered when collection and storage happened. Again, gene expression value may have been measured for some of the biological data-points, but not for others. On the other hand, some measurements may be too expensive to repeat for all data-points; for example, patient's genotype may be measured only if a particular condition holds. All these examples introduce data-point with *missing view*, i.e., all features of a view for a data-point can be missing simultaneously.

Incomplete View On the other hand, it can happen that, we have partial information in a view. As shown in Fig. 1.2 (bottom), for each data-point the pictures from all views are available but some of them are corrupted or partially missing. This kind of data-points with missing features will be denoted henceforth as data-point with *incomplete view*. For example, some unwanted object might have come in the frame while capturing image in a camera. Then the information corresponding to the main object becomes missing for the position of the unwanted object. A second example can be a feedback data containing various codes indicating lack of information like *Don't know, No comments, etc.* [26]. All these examples introduce data-point with *incomplete view*, i.e., some features of a view for a data-point can be missing.

Neglecting views with missing entries definitely restricts models to exploit useful information collected from that view through the data-points for which feature values corresponding to that view are known. Similarly, neglecting data-points with missing entries reduce the number of training data-points for models. Hence, accurate imputation of missing values is required for better learning.

Although multi-view learning in the presence of missing data has drawn a great amount of attention in the recent past and there are quite a lot research papers on multi-view data completion, but there is no comprehensive introduction and review of possible current research approaches on multi-view data completion. Figure 1.3 describes a simple taxonomy of the current approaches. In the worst case, *missing view* and *incomplete view* can occur simultaneously. In this chapter we will mainly discuss existing methods which exclusively solve the missing values problem in multi-view data-set. These methods were proposed mostly to deal with *missing view*s, i.e., row-wise or/and column-wise missing values (top part of Fig. 1.2) in the data matrices and can also be applied for *incomplete view* case.

Multi-View Data and Kernel Completion In order to apply most of the multi-view learning approaches to data-sets with missing views, one needs to pre-process the data-set first for filling in the missing information. Usually, different views contain semantically related content for a given data-point. Often different views share common characteristics and they can also have view-specific characteristics. This property can be useful to impute the missing views in multi-view data-sets.

The conventional matrix completion algorithms [9, 20] solve the problem of recovery of a low-rank matrix in the situation when most of its entries are missing.

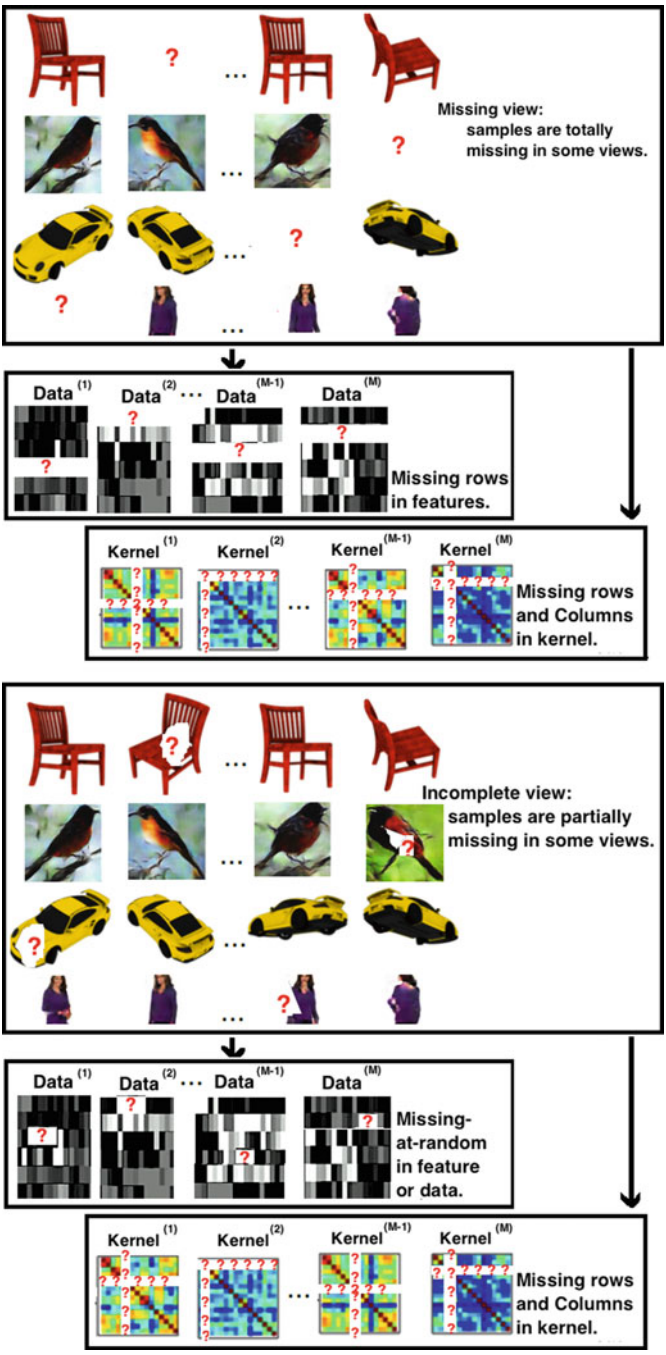


Fig. 1.2 This figure gives a pictorial presentation of two kinds of missing data in multi-view data-sets. The top figure shows view-wise missing data where a data-point is missing in a view and hence the corresponding complete feature row is also missing in data-set. The bottom figure shows that data-points are partially missing in a view and hence create an incomplete view situation in data matrix. But both type of missing values create a missing row and a missing column in the corresponding kernel matrix

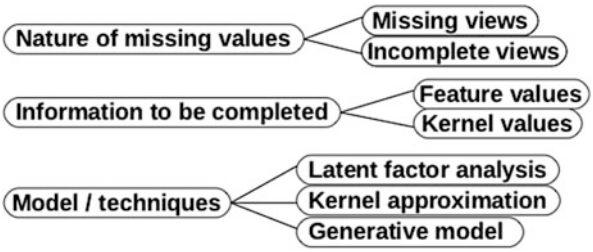


Fig. 1.3 Taxonomy of works in the field of multi-view completion

Missing features in individual views can be imputed using these techniques in case of multi-view data-set with *incomplete view*. But such techniques are meant for imputing missing features for a single-view data-set and unable to exploit the information present in complementary views to complete the missing data in case of multi-view data-set. Hence these methods are out of scope of this chapter.

There are two major approaches to handle missing data problems specifically in multi-view data-set: (1) imputing feature values or data matrices, and (2) imputing kernel matrices. Multi-view data completion methods often complete the data matrices by using *within-view relationships* and *between-view relationship* among data-points.

Ashraphijuo et al. [4] and Lian et al. [23] present a method to impute missing data in feature matrices using matrix factorization-based technique. They learn a factor matrix common for all views and view-specific factor matrix for each view along with factor's weight loadings. This is applicable for multi-view data-set with both *incomplete view* and *missing view*.

Multiple kernel learning (MKL) framework [17] is another efficient approach to accumulate information from multiple sources where no explicit feature information is required but multi-view data is accessed through multiple kernels defined on them. In MKL, the kernel matrices built on features from individual views are combined for better learning. MKL needs to know full kernel matrices for each view. Hence it is enough to impute missing values in kernel matrices instead of imputing missing values in feature space. For both kind of missing information, i.e., *incomplete views* and *missing views*, the missing values for a data-point in a view result into a missing row and a missing column in the kernel matrix of that view.

Existing methods for kernel completion use various assumptions, for example, the presence of a complete kernel matrix in at least one view [30, 31], or all kernel matrices are allowed to be incomplete [6, 23], kernels have homogeneous [27, 30, 31] or heterogeneous eigen structures [6, 23], linear [23] or non-linear approximation [6]. Based on different assumptions, various methods have been developed for kernel completion. In this chapter we will discuss the following methods: (1) matrix factorization-based methods [6], (2) expectation maximization-based methods [27] and (3) generative model-based methods [23]. Moreover, Lian et al. [23] complete the missing values in data and kernel simultaneously.

Similar to multi-view data completion methods, the multi-view kernel completion methods also complete the missing rows and columns in kernel matrices by

learning some within-view and between-view relationships among kernels. Bhadra et al. [6] proposed to learn a relationship among kernel values directly while [31] proposed to learn relationship among eigen structures, and Lian et al. [23] and Bhadra et al. [6] proposed to learn relationship through global component factor matrices for all views. In addition to between-view relationship, Lian et al. [23] and Bhadra et al. [6] learn low rank approximation of view-specific kernel matrices for better generalization which can be considered as learning within-view relationships.

In this chapter we will discuss both imputation of missing values in data matrices and kernel matrices. There are methods for multi-view learning with incomplete views which learn from partially observed data without data completion. One such example is Williams and Carin [34] which does not complete the individual incomplete kernel matrices but complete only the aggregated kernel matrices when all kernels are Gaussian. Those kind of work is out of scope of this chapter.

The remainder of the chapter is organized as follows. In Sect. 1.2, we formulate the problems for missing view and incomplete view setup for both feature matrices and kernel matrices. We also discuss the challenges in missing view problems in the multi-view learning. Section 1.3 presents the state-of-the-art methods for completing missing data in multi-view learning and Sect. 1.4 presents the state-of-the-art multi-view kernel completion methods.

1.2 Incomplete Views vs Missing Views

We assume N data-points $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ from a multi-view input space $\mathscr{X} = \mathscr{X}^{(1)} \times \cdots \times \mathscr{X}^{(M)}$, where $\mathscr{X}^{(m)}$ is the input space generating the mth view. We denote by

$$\mathbf{X}^{(m)} = \{\mathbf{x}_1^{(m)}, \ldots, \mathbf{x}_N^{(m)}\}, \ \forall\ m = 1, \ldots, M,$$

the set of data-points for the mth view, where $\mathbf{x}_i^{(m)} \in \mathscr{X}^{(m)}$ is the ith observation in the mth view and $\mathscr{X}^{(m)}$ is the input space.

Incomplete Views In incomplete view setup some feature values are missing at random in a view. For the tth data-point $\mathbf{x}_t^{(m)}$ of the mth view, there can be a group of features, indexed by J, are missing, i.e., $\mathbf{x}_{t,J}^{(m)}$ are missing. Figure 1.4a shows a pictorial presentation of missing features in incomplete view setup in data matrices of M views, where the J feature set in tth row in $\mathbf{X}^{(1)}$ is missing.

Missing Views In missing view setup, a data-point is completely present or completely missing in a view. Hence in this case only a subset of data-points is completely observed in each view, and correspondingly, a subset of views is observed for each data-point. Let $I_N = [1, \ldots, N]$ be the set of indices of all data-points and $I^{(m)}$ be the set of indices of all available data-points in the mth view. Hence for each view, only a data sub-matrix ($\mathbf{X}_{I^{(m)}}^{(m)}$) corresponding to the rows indexed by $I^{(m)}$ is observed and other rows are missing, i.e., $\mathbf{X}_{I_N/I^{(m)}}^{(m)}$ are missing.

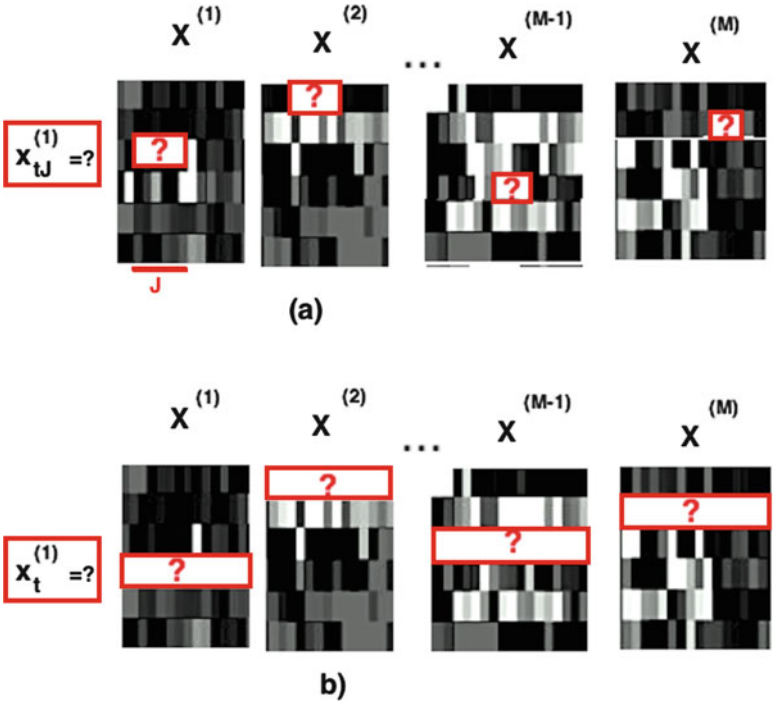


Fig. 1.4 We assume N data-points with M views. (**a**) For incomplete view setup few feature values of some data-points are missing from each individual view and (**b**) for missing view setup few data-points are completely missing in some views and consequently corresponding rows are missing (denoted by '?') in data matrices ($\mathbf{X}^{(m)}$)

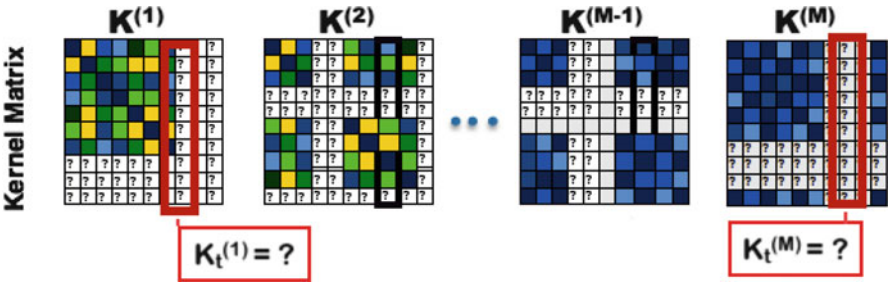


Fig. 1.5 We assume N data-points with M views, with a few data-points missing from each individual view, and consequently corresponding rows and columns are missing (denoted by '?') in kernel matrices ($\mathbf{K}^{(m)}$)

Figure 1.4b shows a pictorial presentation of missing data-points in missing view setup in data matrix of M views, where the tth row in $\mathbf{X}^{(1)}$ is missing.

Missing Rows and Columns in Kernel In this situation a kernel matrix has a complete missing row and a complete missing corresponding column.

Considering an implicit mapping of the observations of the mth view to an inner product space $\mathscr{F}^{(m)}$ via a mapping $\phi^{(m)} : \mathscr{X}^{(m)} \rightarrow \mathscr{F}^{(m)}$, and following the usual recipe for kernel methods [5], we specify the kernel as the inner product in $\mathscr{F}^{(m)}$. The kernel value between the ith and the jth data-points is defined as $k_{ij}^{(m)} = \langle \phi_i^{(m)}, \phi_j^{(m)} \rangle$, where $\phi_i^{(m)} = \phi^{(m)}(\mathbf{x}_i^{(m)})$ and $k_{ij}^{(m)}$ is an element of $\mathbf{K}^{(m)}$, the kernel Gram matrix for the set $\mathbf{X}^{(m)}$.

To define an element $k_{ij}^{(m)}$ in general, one needs complete feature vectors $\mathbf{x}_i^{(m)}$ and $\mathbf{x}_j^{(m)}$ and hence $k_{ij}^{(m)}$ has to be considered as missing kernel values if any of the data-point $\mathbf{x}_i^{(m)}$ or $\mathbf{x}_j^{(m)}$ is partially or completely missing. If the ith data-point has any missing values in the mth view (in both incomplete view and missing view cases), the ith row and the ith column of the kernel matrix $\mathbf{K}^{(m)}$ cannot be calculated and will be missing. Hence while dealing with multi-view kernel completion all existing methods consider missing view setup and hence assume that a subset of data-points is observed in each view, and correspondingly, a subset of views is observed for each data-point. Let $I_N = [1, \ldots, N]$ be the set of indices of all data-points and $I^{(m)}$ be the set of indices of all available data-points in the mth view. Hence for each view, only a kernel sub-matrix ($\mathbf{K}^{(m)}_{I^{(m)} I^{(m)}}$) corresponding to the rows and columns indexed by $I^{(m)}$ is observed. Hence $\mathbf{K}^{(m)}$ has one observed block and 3 unobserved block as follows:

$$\mathbf{K}^{(m)} = \begin{bmatrix} \mathbf{K}^{(m)}_{I^{(m)}, I^{(m)}} & \mathbf{K}^{(m)}_{I^{(m)}, I_N / I^{(m)}} = ? \\ \mathbf{K}^{(m)^T}_{I^{(m)}, I_N / I^{(m)}} = ? & \mathbf{K}^{(m)}_{I_N / I^{(m)}, I_N / I^{(m)}} = ? \end{bmatrix}.$$

Figure 1.5 shows a pictorial presentation of kernel matrices of M views, where the tth row and column of $\mathbf{K}^{(1)}$ is missing.

1.3 Multi-View Data Completion

In this section we will review some recent methods for imputing missing data in missing-view setting. By imputing data we will restrict ourselves to imputing feature vectors. We will discuss methods for kernel completion in the next section. Recall that in case of missing view data-sets, only a data sub-matrix ($\mathbf{X}^{(m)}_{I^{(m)}}$) corresponding to the rows indexed by $I^{(m)}$ is observed and other rows ($\mathbf{X}^{(m)}_{I_N / I^{(m)}}$) are completely missing (Fig. 1.4b).

There are three major approaches in missing-view setting:

- CCA-based robust multi-view data completion ($\text{CCA}_{\text{based}}$) by Subramanya et al. [28, 29]
- Multi-view learning with features and similarities (MLFS) by Lian et al. [23]
- Rank constrained multi-view data completion (LowRank) by Ashraphijuo et al. [4]

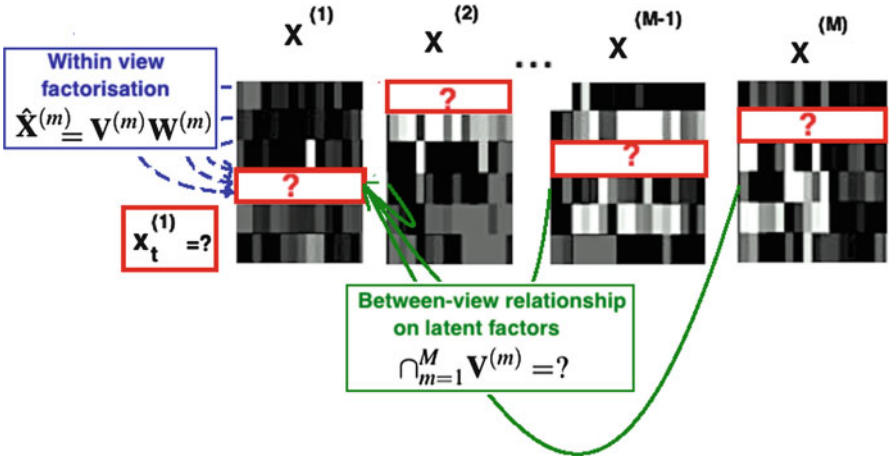


Fig. 1.6 We assume N data-points with M views, with a few data-points missing from each individual view, and consequently corresponding rows and columns are missing (denoted by '?') in data matrices ($\mathbf{X}^{(m)}$). Existing methods predict the missing rows/columns (e.g., the tth column in views 1) with the help of other data-points of the same view (within-view factorization, blue arrows) and the corresponding data-point in other views (relationship among latent factors, green arrows)

Note that, although the above-mentioned feature completion methods have been originally proposed for imputing data in case of missing views, they can be applied to impute partially missing data-points for a view too. All the state-of-the-art multi-view data completion methods predict missing feature vectors $\hat{\mathbf{x}}_t^{(m)}$ by assuming the features for each data-point across different views are generated from a total or partially shared latent factors along with some view-specific factor weights. LowRank [4] assumes that the number of shared and non-shared latent factors for various views is known a priori. On the other hand, $\text{CCA}_{\text{based}}$ and MLFS [23, 29] do not make such assumption and learn the number of latent factors. MLFS was originally proposed for completing data when number of views is greater than two. On the other hand, both $\text{CCA}_{\text{based}}$ and LowRank are mainly applicable for two-view data-set but can be extended easily for more than two views.

All three models, while predicting unknown $\hat{\mathbf{x}}_t^{(m)}$, learn the *within-view factorization* and the relationships among the latent factors across views (Fig. 1.6).

1.3.1 Within-View Factorization

A multi-view data consisting of N data-points and M views can be denoted by an $N \times D_m$ matrix as $\mathbf{X}^{(m)} \in \mathbb{R}^{N \times D_m}$. Let the latent factors be denoted by a matrix $\mathscr{V} = \{\mathbf{V}^{(m)} \in \mathbb{R}^{N \times D_m}\}_{m=1}^{M}$ (where r_m denotes the number of latent factors of data in the

mth view) with view-specific factor loading matrices $\mathscr{W} = \{\mathbf{W}^{(m)} \in \mathscr{R}^{r_m \times D_m}\}_{m=1}^{M}$. Hence the predicted data matrix $\hat{\mathbf{X}}^{(m)}$ can be factorized as:

$$\hat{\mathbf{X}}^{(m)} = \mathbf{V}^{(m)}\mathbf{W}^{(m)}. \tag{1.1}$$

All the existing methods add an additional constraint on $\hat{\mathbf{X}}^{(m)}$ so that the reconstructed feature values closely approximate the known or observed parts of the data. To this end, a loss function measuring the within-view approximation error for each view is defined as:

$$\text{Loss}_{\text{data}}^{(m)} = \|\hat{\mathbf{X}}_{I^{(m)}}^{(m)} - \mathbf{X}_{I^{(m)}}^{(m)}\|_2^2 = \|\mathbf{V}_{I^{(m)}}^{(m)}\mathbf{W}^{(m)} - \mathbf{X}_{I^{(m)}}^{(m)}\|_2^2, \tag{1.2}$$

where $\mathbf{V}_{I^{(m)}}^{(m)}$ denotes the rows of $\mathbf{V}^{(m)}$ indexed by $I^{(m)}$.

1.3.2 Between-View Relation on Latent Factors

When data for a data-point is missing in a view, there is not enough information within that view to impute the features for that data-point. One needs to use other information sources, i.e., the other views where the corresponding features are known. All the existing methods use some kinds of inter-view or between-view relationships on latent factors in order to achieve this. We discuss the different types of relationships below.

Totally Shared Latent Factors According to the most simple setting, $\text{CCA}_{\text{based}}$ [28, 29] assumes that latent factors for all the views are exactly same, i.e., there is a common latent factor $\mathbf{V}$ such that

$$\mathbf{V}^{(m)} = \mathbf{V}.$$

On top of it, $\text{CCA}_{\text{based}}$ does not predict a new feature vector but only selects the nearest neighbour of it from the set of available feature vectors in a view.

Totally Shared and View-Specific Latent Factors Realize that different views may also capture different view-specific aspects of a data-point along with capturing the common aspects that are present in all views. Therefore Ashraphijuo et al. [4] assume that each $\mathbf{V}^{(m)}$ contains some view-specific factors along with some global common factors $\mathbf{V}$ and solves the following problem:

$$\underset{\mathbf{V},\mathbf{V}^{(m)},\mathbf{W}^{(m)}}{\arg\min} \sum_{m=1}^{M} \|[\mathbf{V}\mathbf{V}^{(m)}]_{I^{(m)}}\mathbf{W}^{(m)} - \mathbf{X}_{I^{(m)}}^{(m)}\|_2^2. \tag{1.3}$$

They also assume that the number of common factors and also the number of view-specific factors present in each view is known or given.

Partially Shared Latent Factors MLFS [23] considers that along with the totally shared factors, a subset of views may have a set of additional shared factors. According to Fig. 1.7, the factor $V^{(13)}$ is shared by the first and the third views. To impose this structure on the learned latent factor matrices, MLFS learns a global latent factor matrix $\mathbf{V} \in \mathbb{R}^{N \times r}$ and the associated sparse factor loading matrices $\mathbf{W} = \{\mathbf{W}^{(m)} \in \mathscr{R}^{r \times D_m}\}_{m=1}^{M}$. In addition to that, a structured-sparsity constraint is imposed in the factor loading matrices $\{\mathbf{W}^{(m)}\}_{m=1}^{M}$ of all views, such that some of the rows in these matrices share the same support for non-zero entries, whereas some rows are non-zero only for a subset of these matrices. Figure 1.7 summarizes their basic framework. As a result, it solves the following optimization problem:

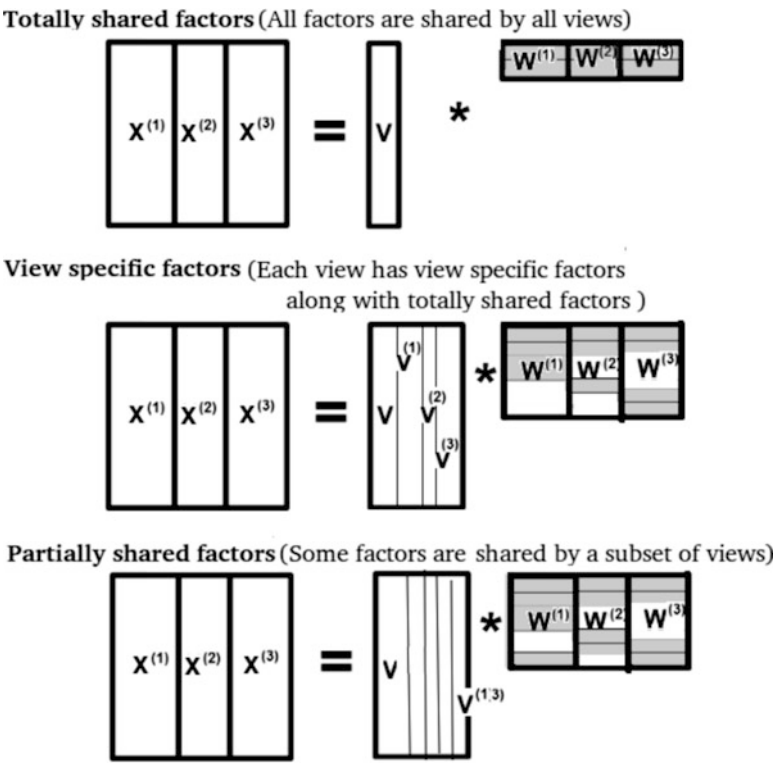


Fig. 1.7 Various kind of sharing of latent factors in existing models are depicted in this figure. $\text{CCA}_{\text{based}}$ [29] assumes all latent factors to be shared by all views, LowRank [4] assumes that along with shared factor each view also has some view-specific factors ($V^{(1)}$ factor is only for the first view) and MLFS [23] in addition to that assumes some factors are also partially shared by only a subset of views ($V^{(13)}$ factor is shared by the first and the third views but not by the second view)

$$\underset{\mathbf{V},\mathbf{W}^{(m)}}{\arg\min} \sum_{m=1}^{M} \|\mathbf{V}_{I^{(m)}}\mathbf{W}^{(m)} - \mathbf{X}^{(m)}_{I^{(m)}}\|_2^2$$

$$\text{with row sparsity constraints on } \mathbf{W}^{(m)}, \quad \forall m = 1, \ldots, M. \tag{1.4}$$

MLFS uses generative model to learn the latent factors and also to predict the missing features. It uses group-wise automatic relevance determination [32] as the sparsity inducing prior on $\{\mathbf{W}^{(m)}\}_{m=1}^{M}$, which also helps in inferring r by shrinking the unnecessary rows in $\mathbf{W}$ to near zero.

1.4 Multi-View Kernel Completion

Multi-view kernel completion techniques predict a complete positive semi-definite (PSD) kernel matrix ($\hat{\mathbf{K}}^{(m)} \in \mathbb{R}^{N\times N}$) corresponding to each view where some rows and columns of the given kernel ($\mathbf{K}^{(m)} \in \mathbb{R}^{N\times N}$) are fully missing (Fig. 1.5). The crucial task is to predict the missing (tth) rows and columns of $\hat{\mathbf{K}}^{(m)}$, for all $t \in I_h^{(m)} = \{I_N / I^{(m)}\}$. Existing methods for kernel completion have addressed completion of multi-view kernel matrices with various assumptions. In this section, we will discuss details of the following state-of-the-art methods for multi-view kernel completion.

- Multi-view learning with features and similarities (MLFS) by Lian et al. [23]
- Multi-view kernel completion (MKC_{sdp}, MKC_{app}, $\text{MKC}_{\text{embd(ht)}}$, $\text{MKC}_{\text{embd(hm)}}$) by Bhadra et al. [6]
- The expectation maximization-based method (EM_{based}) by Tsuda et al. [31] and Shao et al. [27]

These methods have various assumptions, i.e., any auxiliary full kernel matrix is available or not; all kernel functions are linear or non-linear; kernel functions in different views are homogeneous or heterogeneous in nature, etc. The application/assumption of these methods is described in Table 1.1.

Almost all existing state-of-the-art approaches for predicting $\hat{\mathbf{K}}^{(m)}$ are based on learning both between-view and within-view relationships among the kernel values (Fig. 1.8). The sub-matrix $\hat{\mathbf{K}}^{(m)}_{I^{(m)}I^{(m)}}$ should be approximately equal to the observed matrix $\mathbf{K}^{(m)}_{I^{(m)}I^{(m)}}$. However, in some approaches such as [6, 23, 31], approximation quality of the two parts of the kernel matrices is traded.

1.4.1 Within-View Kernel Relationships

For being a valid kernel the predicted $\hat{\mathbf{K}}^{(m)}$ needs to be a positive semi-definite matrix for each m. The expectation maximization-based method (abbreviated

Table 1.1 Table shows the assumption made by the state-of-the-art multi-view kernel completion methods and the algorithm used by them

	Methods					
	MKC_{sdp} [6]	MKC_{app} [6]	EM_{based} [27, 31]	$MKC_{embd(hm)}$ [6]	MLFS [23]	$MKC_{embd(ht)}$ [6]
Properties of missing values						
Some rows and columns of a kernel matrix are missing entirely	Yes	Yes	Yes	Yes	Yes	Yes
Views are homogeneous	Yes	Yes	Yes	Yes	Yes	Yes
No need of at least one complete kernel	Yes	Yes	x	Yes	Yes	Yes
Kernels are spectral variant of each other	x	x	Yes	Yes	Yes	Yes
Views are heterogeneous	x	x	x	x	Yes	Yes
Highly non-linear kernel	Yes	x	x	x	x	Yes
Algorithm						
Semi-definite programming	Yes	x	x	x	x	x
Non-convex optimization	x	Yes	x	Yes	x	Yes
Generative model	x	x	x	x	Yes	x
Expectation maximization	x	x	Yes	x	x	x

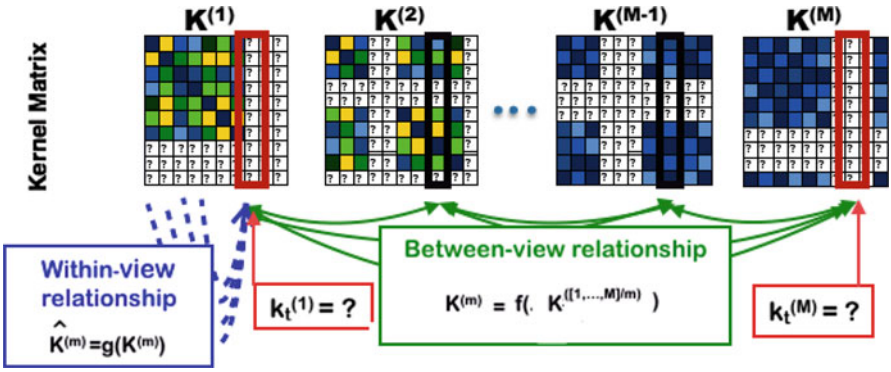


Fig. 1.8 We assume N data-points with M views, where a few data-points missing from each individual view, and consequently corresponding rows and columns are missing (denoted by '?') in kernel matrices ($\mathbf{K}^{(m)}$). The proposed methods predict the missing kernel rows/columns (e.g., the tth column in views 1 and m) with the help of other data-points of the same view (within-view relationship, blue arrows) and the corresponding data-point in other views (between-view relationship, green arrows)

henceforth as EM_{based}) [31] does not assume any other relationship among data-points but only guarantees the PSD property of individual kernel matrix. In optimization problem, it requires explicit positive semi-definiteness constraints:

$$\hat{\mathbf{K}}^{(m)} \succeq 0, \tag{1.5}$$

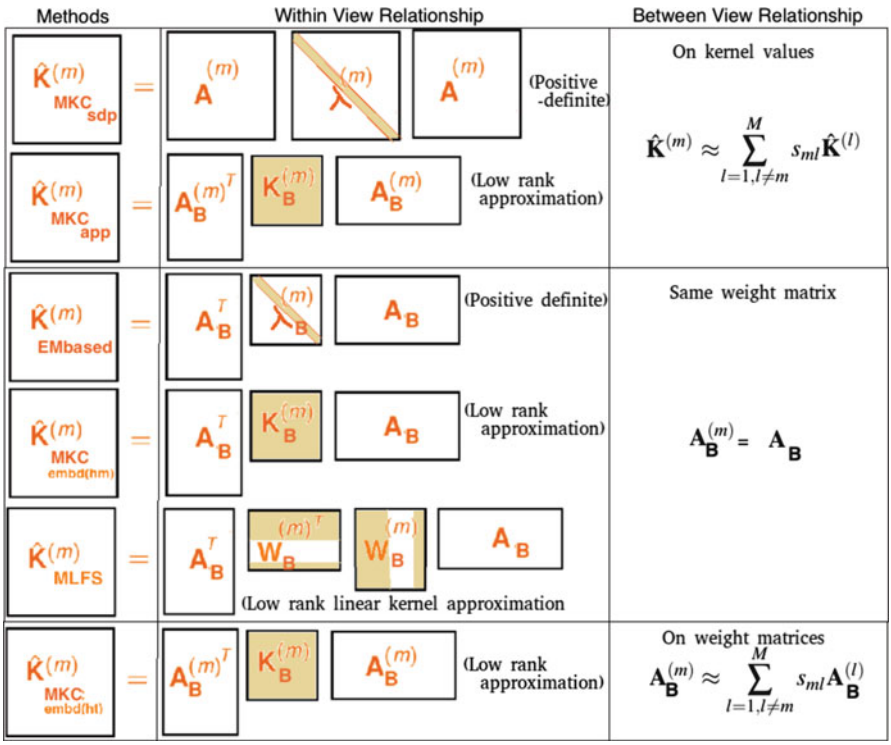


Fig. 1.9 Within-view and between-view relationships in kernel values assumed in different models

to guarantee the PSD property of individual kernel matrix. MKC [6] has also considered no other but only explicit PSD constraint in one of its formulation denoted as MKC_{sdp}.

But with missing rows and columns these constraints leave the model too flexible. Hence some methods also predict missing rows/columns by learning some other relationship $\hat{\mathbf{k}} = g(\mathbf{K})$ among data-points in the same view. Relationship among data-points of same view is known as *within-view relationship*. For within-view relationship learning, some methods rely on the concept of low rank kernel approximation [23], while others use the concept of local linear embedding [6]. Figure 1.9 pictorially presents the within-view relationship considered in various proposed methods.

Multi-view learning with features and similarities [23] assume kernels are linear in nature and hence can have low rank approximation, i.e.,

$$\hat{\mathbf{K}}^{(m)} = g(\mathbf{K}^{(m)}) = \mathbf{U}^{(m)^T}\mathbf{U}^{(m)} = \mathbf{A}^T\mathbf{W}^{(m)^T}\mathbf{W}^{(m)}\mathbf{A}, \tag{1.6}$$

where $\mathbf{U}^{(m)} \in \mathscr{R}^{N\times r}$ with $r << N$. $\mathbf{A}$ and $\mathbf{W}^{(m)}$ are denoted, respectively, as global latent factor matrices and the associated factor loading matrices for the mth view. A by-product of this approximation is that in optimization, PSD property is automatically guaranteed without inserting explicit positive semi-definiteness constraints.

Recently proposed multi-view kernel completion [6] considered retaining the non-linear property of a kernel by assuming local embedding in feature space instead of kernel space. It assumes that in each view there exists a sparse embedding in $\mathscr{F}$, given by a small set of data-points $B^{(m)} \subset I^{(m)}$, called a basis set, that is able to represent all possible feature representations in that particular view. Hence MKC reconstructs the feature map of tth data-point $\phi_t^{(m)}$ by a sparse linear combination of a subset of observed data-points

$$\hat{\phi}_t^{(m)} = \sum_{i\in B^{(m)}} a_{it}^{(m)} \phi_i^{(m)}, \tag{1.7}$$

where $a_{it} \in \mathbb{R}$ is the reconstruction weight of the ith feature representation for the tth data-point. Hence, approximated kernel values can be expressed as

$$\hat{k}_{tt'}^{(m)} = \langle \hat{\phi}_t^{(m)}, \hat{\phi}_{t'}^{(m)} \rangle = \sum_{i,j\in I^{(m)}} a_{it}^{(m)} a_{jt'}^{(M)} \langle \phi_i^{(m)}, \phi_j^{(m)} \rangle. \tag{1.8}$$

The above formulation retains the non-linearity of the feature map ϕ and the corresponding kernel. $\mathbf{A} = \left(a_{ij}\right)_{i,j=1}^{N}$ is the matrix of re-constructing weights. Further, $\mathbf{A}_{B^{(m)}}^{(m)}$ is the sub-matrix of $\mathbf{A}$ containing the rows indexed by $B^{(m)}$, the set of basis vectors in kernel feature space in view. Thus the reconstructed kernel matrix $\hat{\mathbf{K}}$ can be written as

$$\hat{\mathbf{K}}^{(m)} = g(\mathbf{K}^{(m)}) = \mathbf{A}_{B^{(m)}}^{(m)^T} \mathbf{K}_{B^{(m)}} \mathbf{A}_{B^{(m)}}^{(m)}. \tag{1.9}$$

Note that $\hat{\mathbf{K}}$ is positive semi-definite when $\mathbf{K}B^{(m)}$ is positive semi-definite. Thus, a by-product of this approximation is that in optimization PSD property is automatically guaranteed without inserting explicit positive semi-definiteness constraints.

Intuitively, the reconstruction weights are used to extend the known part of the kernel to the unknown part, in other words, the unknown part is assumed to reside within the span of the known part.

To select a sparse set of reconstruction weights, Bhadra et al. [6] regularizes the reconstruction weights by the $\ell_{2,1}$ norm [3] of the reconstruction weight matrix,

$$\|\mathbf{A}^{(m)}\|_{2,1} = \sum_i \sqrt{\sum_j (a_{ij})^2}. \tag{1.10}$$

Finally, for the observed part of the kernel, all existing methods add the additional constraint on $\hat{\mathbf{K}}^{(m)}$ that the reconstructed kernel values closely approximate the known or observed values. To this end, a loss function measuring the within-view approximation error for each view is defined as

$$\text{Loss}^{(m)}_{\text{within}} = \|g(\mathbf{K}^{(m)})_{I^{(m)}I^{(m)}} - \mathbf{K}^{(m)}_{I^{(m)}I^{(m)}}\|_2^2. \tag{1.11}$$

Hence, for individual views the observed part of a kernel is approximated by $g(\mathbf{K}^{(m)})_{I^{(m)}I^{(m)}}$ by optimizing the parameters $(\mathbf{A}, \mathbf{W})$ (Eqs. (1.10) and (1.11)) in

$$\arg\min_{\mathbf{A},\mathbf{W}} \|g(\mathbf{K}^{(m)})_{I^{(m)}I^{(m)}} - \mathbf{K}_{I^{(m)}I^{(m)}}\|_2^2 + \lambda\|\mathbf{A}^{(m)}\|_{2,1}, \tag{1.12}$$

where λ is a user-defined hyper-parameter which indicates the weight of regularization.

Without the $\ell_{2,1}$ regularization, the above approximation loss could be trivially optimized by choosing $\mathbf{A}^{(m)}$ as the identity matrix. The $\ell_{2,1}$ regularization will have the effect of zeroing out some of the diagonal values and introducing non-zeroes to the sub-matrix $\mathbf{A}^{(m)}_B$, corresponding to the rows indexed by B, where $B = \{i | a_{ii} \neq 0\}$.

MKC [6] also claimed that Eq. (1.12) corresponds to a generalized form of the Nyström method [35] which is a sparse kernel approximation method that has been successfully applied to efficient kernel learning. Nyström method finds a small set of vectors (not necessarily linearly independent) spanning the kernel, whereas MKC method searches for linearly independent basis vectors and optimizes the reconstruction weights for the data-points.

1.4.2 Between-View Kernel Relationships

For a completely missing row or column of a kernel matrix, there is not enough information available for completing it within the same view, and hence the completion needs to be based on other information sources, i.e., the other views where the corresponding kernel parts are known. All existing methods learn some kind of inter-view or between-view relationships $f(\cdot)$ to predict missing values in a view with help of information present in other views, i.e., $\hat{\mathbf{K}}^{(m)} \approx f(\hat{\mathbf{K}}^{[1,\ldots,M]/m})$.

Various existing methods learn different kind of inter-view or between-view relationships among kernels to predict missing values depending upon various assumptions regarding the available information. This gives us between-view loss as:

$$\text{Loss}^{(m)}_{\text{between}}(\hat{\mathbf{K}}, f(\hat{\mathbf{K}}^{[1,\ldots,M]/m})) = \|\hat{\mathbf{K}}^{(m)} - f(\hat{\mathbf{K}}^{[1,\ldots,M]/m})\|_2^2. \tag{1.13}$$

The MKC_{app} [6] learns linear relationships among values of each element of kernel matrices (k_{ij}) based on learning a convex combination of the kernels, extending the multiple kernel learning [2, 12] techniques to kernel completion. The second technique discussed in EM_{based} [27, 31] learns relationship among eigen structure of kernels of various views. In case kernels of different views are heterogeneous and do not share the similar eigen-spectra, the third technique discussed in MLFS [23] and MKC [6] learns the linear relationship among the latent low-dimensional embedding of different views based on learning reconstruction weights so that they share information among all views [23] or selected set of similar views [6]. Figure 1.9 presents summary of the between-view relationship considered in various proposed method.

Between-View Learning of Kernel Values In multi-view kernel completion perhaps the simplest situation arises when the kernels of the different views are similar. In that case to learn between-view relationships MKC_{sdp} and MKC_{app} [6] express the individual kernel matrix corresponding to each view as a convex combination of kernel matrices of the other views. Hence the proposed model learns kernel weights $\mathbf{S} = (s_{ml})_{m,l=1}^{M}$ between all possible pairs of kernels (m, l) such that

$$\hat{\mathbf{K}}^{(m)} \approx f(\cdot) = \sum_{l=1, l\neq m}^{M} s_{ml}\hat{\mathbf{K}}^{(l)}, \tag{1.14}$$

where the kernel weights are confined to a convex combination as:

$$\mathscr{S} = \left\{ \mathbf{S} | s_{ml} \geq 0, \sum_{l=1, l\neq m}^{M} s_{ml} = 1 \right\}. \tag{1.15}$$

The kernel weights then can flexibly pick up a subset of relevant views to the current view m.

Between-View Learning Assuming Rotated Eigen Structure of a Known Kernel In practical applications, the kernels arising in a multi-view setup might be very heterogeneous in their distribution. In such cases, it might not be realistic to find a convex combination of other kernels that are closely similar to the kernel of a given view. In particular, when the $\hat{\mathbf{K}}^{m}$ is assumed to be approximated by the parametric model which is derived from the spectral variants of $\mathbf{K}^{(O)}$, which is an $N \times N$ fully observed auxiliary kernel matrix [27, 31] the missing values in $\hat{\mathbf{K}}^{m}$ is completed by expressing the kernel matrix as the spectral variants of $\mathbf{K}^{(O)}$, i.e.,

$$\hat{\mathbf{K}}^{(m)} \approx f(\cdot) = \sum_{i=1}^{N} s_i^{(m)} M_i, \text{ where } M_i = v_i v_i^T, \tag{1.16}$$

where $s_i^{(m)} \geq 0$ and the eigen decomposition of $\mathbf{K}^{(O)}$ is denoted as $\mathbf{K}^{(O)} = \sum_{i=1}^{N} \lambda_i v_i v_i^T$ with λ_i's and v_i's as eigen values and eigen vectors, respectively. Here all incomplete kernels considered to have eigen vectors same as that of $\mathbf{K}^{(O)}$, but their own set of eigen values ($s_i^{(m)}$'s).

Between-View Learning of Reconstruction Weights When the eigen vectors of a kernel matrix are rotated from the eigen vectors of kernels of other views, Bhadra et al. [6] propose an alternative approach $\text{MKC}_{\text{embd(hm)}}$, where instead of the kernel values or eigen vectors, they assume that the basis sets and the reconstruction weights (Eq. (1.9)) have between-view dependencies as follows:

$$\mathbf{A}^{(1)} = \ldots = \mathbf{A}^{(M)}. \tag{1.17}$$

The reconstructed kernel is thus given by

$$\hat{\mathbf{K}}^{(m)} \approx f(\cdot) = \mathbf{A}^T \mathbf{K}^{(m)}_{B^{(m)}} \mathbf{A}. \tag{1.18}$$

However, assuming that kernel functions in *all* the views have similar eigen-vectors is also unrealistic for many real-world data-sets with heterogeneous sources and kernels applied to them. On the contrary, it is quite possible that only for a subset of views the eigen-vectors of approximated kernel are linearly related. $\text{MKC}_{\text{embd(ht)}}$ [6] thus allow the views to have different reconstruction weights, but a parametrized relationship among them, i.e., the reconstruction weights in a view can be approximated by a convex combination of the reconstruction weights of other views:

$$\mathbf{A}^{(m)} \approx \sum_{l=1, l\neq m}^{M} s_{ml} \mathbf{A}^{(l)}, \tag{1.19}$$

where the coefficients s_{ml} are defined as in Eq. (1.14).

The reconstructed kernel is thus given by

$$\hat{\mathbf{K}}^{(m)} \approx f(\cdot) = \left(\sum_{l=1, l\neq m}^{M} s_{ml} \mathbf{A}^{(l)^T}_{B^{(m)}} \right) \mathbf{K}^{(m)}_{B^{(m)}} \left(\sum_{l=1, l\neq m}^{M} s_{ml} \mathbf{A}^{(l)}_{B^{(m)}} \right). \tag{1.20}$$

This also allows the model to find an appropriate set of related kernels from the set of available incomplete kernels, for each missing entry.

1.4.3 Optimization Methods

Almost all state-of-the-art multi-view kernel completion methods optimize their parameters by minimizing both within-view and between-view loss, i.e., using Eqs. (1.11) and (1.13)

$$\arg\min_{\mathbf{A},\mathbf{W}} \text{Loss}^{(m)}_{\text{between}}(\hat{\mathbf{K}},\, f(\hat{\mathbf{K}}^{[1,\ldots,M]/m})) + \lambda_2 \sum_{m=1}^{M} \text{Loss}^{(m)}_{\text{within}} \tag{1.21}$$

Like their assumptions and formulation various methods also use various optimization techniques to solve their optimization formulation.

- In multi-view setting, MLFS [23] proposed a generative model-based method which approximates the similarity matrix for each view as a linear kernel in some low-dimensional space.
- MKC [6] solves their non-convex problem with the help of solving sequence of convex problems using proximal gradient method.
- On the other hand, EM_{based} [27, 31] have proposed an expectation maximization-based method to complete an incomplete kernel matrix for a view, with the help of a complete kernel matrix from another view.

1.5 Discussion

This section discusses the applicability of various methods depending upon characteristic of data-set. The applicability of methods on data or kernel completion depends upon the number and type of missing features and also the degree of heterogeneity in information of different views. Degree of heterogeneity of information can be captured in eigen-spectrum of the kernel matrix [6].

Multi-View Data Completion The imputation in feature matrix is more appropriate if there are less number of missing features for each data-point (incomplete view) or less number of features per view (missing view).

The very first multi-view matrix completion method, $\text{CCA}_{\text{based}}$ uses a heuristics-based method for predicting missing data-points. Assumption of having only common latent factors is also not true for real-world data. This technique is applicable if views are homogeneous. When eigen-spectra of kernel matrices (in this case linear kernel) for all views are similar (the left most plot of Fig. 1.10) then views are considered to be homogeneous and hence can be explained with shared factors alone. In real data, each view mostly has some view-specific complementary information. Hence each view cannot be explained only by shared factors, otherwise prediction accuracy of $\text{CCA}_{\text{based}}$ is not expected to be good enough in real-world data.

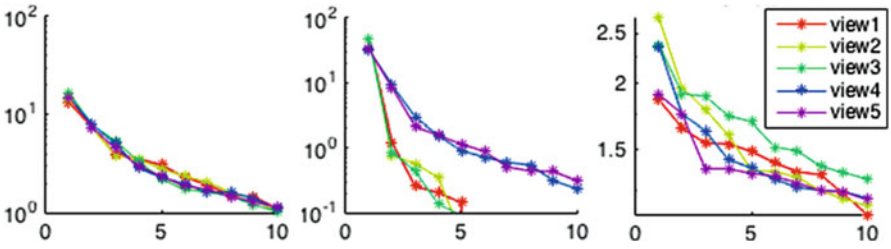


Fig. 1.10 This figure shows eigen-spectra of three data-sets with five views. In each plot x-axis and y- axis denote the k and the kth eigen value, respectively. From left to right the heterogeneity of views increases

On the other hand, when views contain complementary information then the eigen-spectra of views are totally different from each other (like right most plot of Fig. 1.10). In this case learning of any between-view relationship does not help.

Partially shared factors can explain heterogeneous views better, and hence should work well in case of heterogeneous views. Here the eigen-spectra of a subset of views or a part of eigen-spectra of all views are overlapping (like middle plot of Fig. 1.10). Lian et al. [23] have shown that MLFS is capable of predicting missing data with high accuracy when views are heterogeneous in nature, but did not compare their performance with $\text{CCA}_{\text{based}}$. Ashraphijuo et al. [4] have given a thorough theoretical analysis on their model but did not show empirical performance of LowRank on real-world data.

Multi-View Kernel Completion When the final learning model is based on kernel, one can directly impute kernel matrices. The imputation in kernel matrix is more efficient if there are large number of missing features for each data-point (incomplete view) or number of features per view (missing view) is more than the number of data-points.

An extended comparative study among the state-of-the-art methods on a variety of data-sets, with different types of kernel functions in different views, along with different amounts of missing data-points has been reported in [6]. They claim that EM_{based} performs well if at least complete auxiliary kernel is available but fails to perform well if none of the given kernel is complete. Moreover, it performs well when eigen-spectra of incomplete kernels and auxiliary kernels are same.

$\text{MKC}_{\text{embd(hm)}}$ and MLFS also use shared weight matrix $\mathbf{A}$, hence perform well when the underlying sub-spaces, i.e., the eigen-spectra of kernel matrices of all views are same (left most plot of Fig. 1.10). Again, MKC_{app} and $\text{MKC}_{\text{embd(ht)}}$ learn sparse relationships among views considering a subset of views are related to each other. Hence for heterogeneous views, where the eigen-spectra of a subset of view are same, these methods perform well. However, with the increase of non-linearity in kernel values performance of MKC_{app} and MLFS deteriorates. For non-linear kernels, $\text{MKC}_{\text{embd(ht)}}$ and MKC_{sdp} are more appropriate. $\text{MKC}_{\text{embd(ht)}}$ outperforms all other variants of MKC in almost all data-sets, while MKC_{sdp}

outperforms all other approximated methods when kernel functions are highly non-linear.

1.5.1 Case Study: Reuters RCV1/RCV2 Multilingual Data

Reuters RCV1/RCV2 multilingual data-set [1] contains aligned documents for five languages (English, French, Germany, Italian and Spanish). In case some documents are missing in some languages, then this is an example of data-points with missing views. Number of data-points for this data-set is more than a million (precisely 111,740), and hence kernel size is large. Also we see that the eigen-spectra of kernel matrices using the latent semantic kernel [13] for all languages/views show homogeneity among views (Fig. 1.11). Hence $\text{MKC}_{\text{embd(hm)}}$ or MKC_{app} will be appropriate for this data-set. Again $\text{CCA}_{\text{based}}$ method can also be applied after a reduction in number of features in views by some feature selection techniques.

1.5.2 Case Study: Dream Challenge 7 Data-Set (DREAM)

Dream challenge 7 data-set (DREAM) [14, 18] contains genomic characterizations of multiple types on 53 breast cancer cell lines. They consist of DNA copy number variation, transcript expression values, whole exome sequencing, RNA sequencing data, DNA methylation data and RPPA protein quantification measurements. In addition, some of the views are missing for some cell lines.

The number of features in all views is large compared to the number of data-points (25). Hence kernel matrix completion methods need to impute less number of missing values and will be more appropriate. Figure 1.11 shows the eigen-spectra of the kernel matrices (Gaussian kernels on all six views after normalizing the data-

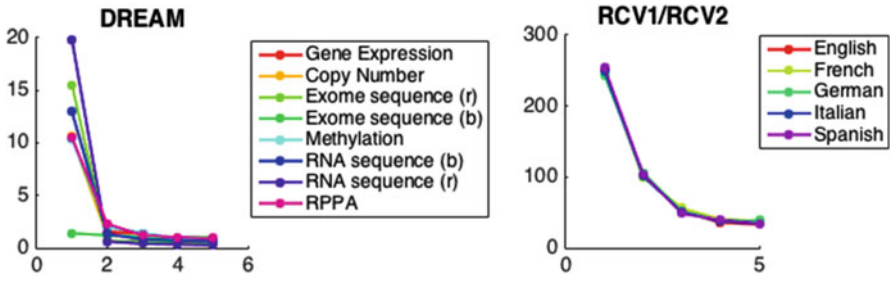


Fig. 1.11 This figure shows eigen-spectra of the Dream challenge 7 and the RCV1/RCV2 data-set. In each plot, x-axis and y-axis denote the k and the kth eigen value, respectively. The heterogeneity of views is more in the Dream challenge 7 data

sets and Jaccard's kernel function over binarized exome data and RNA sequencing data) of all views.

Figure 1.11 shows that the eigen-spectra of the kernel matrices are quite different for different views. Hence we need a method which can deal with heterogeneity like MKC_{app}, $\text{MKC}_{\text{embd(ht)}}$ or MLFS. Also we can see one group of spectra relatively heavy tailed, which is corresponding to highly non-linear kernel matrices. Therefore, it is more appropriate to try MKC_{app} or $\text{MKC}_{\text{embd(ht)}}$.

1.6 Conclusion and Possible Future Direction

In this chapter we present a comprehensive understanding of similarities and dissimilarities of representative as well as newly proposed models for completion of missing information in multi-view data-sets. We focus more on the methods which can deal with missingness where entire feature vector corresponding to some data-points is missing in some views and presents in other views. These methods are also applicable for incomplete view problems too. Existing methods for completing features or kernel values have a common framework of learning using within-view relationships and between-view relationships among data-points. In general their approaches for modeling within-view relationships are similar but they differ in modeling between-view relationships.

All the existing multi-view data or kernel completion methods can predict missing values with high accuracy only if relationship among views are linear in nature. It would be good to explore the possibilities where views are non-linearly related. Recently, a deep factorization-based single view matrix completion method [16] considers non-linear function among data-points in the same view. As a future work in this line of research, it would be interesting to explore the possibility of extending it to multi-view setup. All the existing methods do offline learning and hence the streaming extension of them can show the feasibility of their framework in online learning and active learning.

References

1. Amini, M., Usunier, N., Goutte, C.: Learning from multiple partially observed views - an application to multilingual text categorization. In: Advances in Neural Information Processing Systems, vol. 22, pp. 28–36 (2009)
2. Argyriou, A., Micchelli, C.A., Pontil, M.: Learning convex combinations of continuously parameterized basic kernels. In: Proceedings of the 18th Annual Conference on Learning Theory, pp. 338–352 (2005)
3. Argyriou, A., Evgeniou, T., Pontil, M.: Multi-task feature learning. In: Advances in Neural Information Processing Systems, pp. 41–48 (2006)
4. Ashraphijuo, M., Wang, X., Aggarwal, V.: A characterization of sampling patterns for low-rank multi-view data completion problem. In: 2017 IEEE International Symposium on Information Theory (ISIT), pp. 1147–1151. IEEE (2017)

5. Bach, F., Lanckriet, G., Jordan, M.: Multiple kernel learning, conic duality, and the SMO algorithm. In: Proceedings of the 21st International Conference on Machine Learning, pp. 6–13. ACM, New York (2004)
6. Bhadra, S., Kaski, S., Rousu, J.: Multi-view kernel completion. Mach. Learn. **106**(5), 713–739 (2017)
7. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the Eleventh Annual Conference on Computational Learning Theory, pp. 92–100. ACM, New York (1998)
8. Bunte, K., Leppäaho, E., Saarinen, I., Kaski, S.: Sparse group factor analysis for biclustering of multiple data sources. Bioinformatics **32**(16), 2457–2463 (2016)
9. Candès, E.J., Recht, B.: Exact matrix completion via convex optimization. Found. Comput. Math. **9**(6), 717 (2009)
10. Chao, G., Sun, S.: Multi-kernel maximum entropy discrimination for multi-view learning. Intell. Data Anal. **20**(3), 481–493 (2016)
11. Christoudias, C., Urtasun, R., Darrell, T.: Multi-view learning in the presence of view disagreement (2012). Preprint. arXiv:1206.3242
12. Cortes, C., Mohri, M., Rostamizadeh, A.: Algorithms for learning kernels based on centered alignment. J. Mach. Learn. Res. **13**, 795–828 (2012)
13. Cristianini, N., Shawe-Taylor, J., Lodhi, H.: Latent semantic kernels. J. Intell. Inf. Syst. **18**(2-3), 127–152 (2002)
14. Daemen, A., Griffith, O., Heiser, L., et al.: Modeling precision treatment of breast cancer. Genome Biol. **14**(10), R110 (2013)
15. Dhillon, P., Foster, D.P., Ungar, L.H.: Multi-view learning of word embeddings via CCA. In: Advances in Neural Information Processing Systems, pp. 199–207 (2011)
16. Fan, J., Chow, T.: Deep learning based matrix completion. Neurocomputing **266**, 540–549 (2017)
17. Gönen, M., Alpaydın, E.: Multiple kernel learning algorithms. J. Mach. Learn. Res. **12**(Jul), 2211–2268 (2011)
18. Heiser, L.M., Sadanandam, A., et al.: Subtype and pathway specific responses to anticancer compounds in breast cancer. Proc. Natl. Acad. Sci. **109**(8), 2724–2729 (2012)
19. Kan, M., Shan, S., Zhang, H., Lao, S., Chen, X.: Multi-view discriminant analysis. IEEE Trans. Pattern Anal. Mach. Intell. **38**(1), 188–194 (2016)
20. Klopp, O., Lounici, K., Tsybakov, A.B.: Robust matrix completion. Probab. Theory Relat. Fields **169**(1–2), 523–564 (2017)
21. Kumar, A., Rai, P., Daume, H.: Co-regularized multi-view spectral clustering. In: Advances in Neural Information Processing Systems, pp. 1413–1421 (2011)
22. Li, Y., Wu, F.X., Ngom, A.: A review on machine learning principles for multi-view biological data integration. Brief. Bioinform. (2018). https://doi.org/10.1093/bib/bbw113
23. Lian, W., Rai, P., Salazar, E., Carin, L.: Integrating features and similarities: Flexible models for heterogeneous multiview data. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence, pp. 2757–2763 (2015)
24. Livescu, K., Stoehr, M.: Multi-view learning of acoustic features for speaker recognition. In: IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU 2009, pp. 82–86. IEEE (2009)
25. Müller, K., Schwarz, H., Marpe, D., Bartnik, C., Bosse, S., Brust, H., Hinz, T., Lakshman, H., Merkle, P., Rhee, F.H., et al.: 3d high-efficiency video coding for multi-view video and depth data. IEEE Trans. Image Process. **22**(9), 3366–3378 (2013)
26. Schafer, J.L., Graham, J.W.: Missing data: our view of the state of the art. Psychol. Methods **7**(2), 147 (2002)
27. Shao, W., Shi, X., Yu, P.S.: Clustering on multiple incomplete datasets via collective kernel learning. In: 2013 IEEE 13th International Conference on Data Mining (ICDM), pp. 1181–1186. IEEE (2013)

28. Subramanya, S., Li, B., Liu, H.: Robust integration of multiple information sources by view completion. In: IEEE International Conference on Information Reuse and Integration, IRI 2008, pp. 398–403. IEEE (2008)
29. Subramanya, S., Wang, Z., Li, B., Liu, H.: Completing missing views for multiple sources of web media. Int. J. Data Min. Model. Manag. **1**(1), 23–44 (2008)
30. Trivedi, A., Rai, P., Daumé III, H., DuVall, S.L.: Multiview clustering with incomplete views. In: Proceedings of the NIPS Workshop (2005)
31. Tsuda, K., Akaho, S., Asai, K.: The em algorithm for kernel matrix completion with auxiliary data. J. Mach. Lear. Res. **4**, 67–81 (2003)
32. Virtanen, S., Klami, A., Khan, S., Kaski, S.: Bayesian group factor analysis. In: Artificial Intelligence and Statistics, pp. 1269–1277 (2012)
33. Wan, X.: Co-training for cross-lingual sentiment classification. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1, pp. 235–243. Association for Computational Linguistics (2009)
34. Williams, D., Carin, L.: Analytical kernel matrix completion with incomplete multi-view data. In: Proceedings of the ICML Workshop on Learning With Multiple Views (2005)
35. Williams, C., Seeger, M.: Using the nyström method to speed up kernel machines. In: Proceedings of the 14th Annual Conference on Neural Information Processing Systems, pp. 682–688. No. EPFL-CONF-161322 (2001)
36. Xu, C., Tao, D., Xu, C.: A survey on multi-view learning (2013). Preprint. arXiv:1304.5634
37. Xu, J., Han, J., Nie, F.: Discriminatively embedded k-means for multi-view clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5356–5364 (2016)
38. Xue, Z., Li, G., Huang, Q.: Joint multi-view representation learning and image tagging. In AAAI, pp. 1366–1372 (2016)
39. Zhao, J., Xie, X., Xu, X., Sun, S.: Multi-view learning overview: recent progress and new challenges. Inf. Fusion **38**, 43–54 (2017)

Chapter 2
Multi-View Clustering

Deepak P and Anna Jurek-Loughrey

Abstract With a plethora of data capturing modalities becoming available, the same data object often leaves different kinds of digital footprints. This naturally leads to datasets comprising the same set of data objects represented in different forms, called multi-view data. Among the most fundamental tasks in unsupervised learning is that of clustering, the task of grouping data objects into groups of related objects. Multi-view clustering (MVC) is a flourishing field in unsupervised learning; the MVC task considers leveraging multiple views of data objects in order to arrive at a more effective and accurate grouping than what can be achieved by just using one view of data. Multi-view clustering methods differ in the kind of modelling they use in order to fuse multiple views, by managing the synergies, complimentarities, and conflicts across data views, and arriving at a single clustering output across the multiple views in the dataset. This chapter provides a survey of a sample of multi-view clustering methods, with an emphasis on bringing out the wide diversity in solution formulations that have been considered. We pay specific attention to enable the reader understand the intuition behind each method ahead of describing the technical details of the method, to ensure that the survey is accessible to readers who may not be machine learning specialists. We also outline some popular datasets that have been used to empirically evaluate MVC methods.

2.1 Introduction

Exploratory data analysis is becoming increasingly important, with massive amounts of data being created every moment, vastly outpacing any chance of processing them manually. Modern data scenarios routinely embrace complex objects, whose representations encompass multiple forms—often called *views*—possibly containing even different types of data, such as text, images, sets, and

Deepak P (✉) · A. Jurek-Loughrey
Queen's University Belfast, Belfast, UK
e-mail: deepaksp@acm.org; a.jurek@qub.ac.uk

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_2

sequences. As a simple example, our social media streams are often occupied by a mix of text (which itself appears in multiple forms), images, and videos. Archeological artifacts are often represented electronically using their geo-location, their 3D view, and the properties of their ingredients. The extravagance of data is best pronounced in fields that involve constant monitoring such as astronomy where time series data of incoming radiation is incessantly captured along with other sensoring modalities. The emergence of persuasive health technologies such as activity trackers that hold a variety of sensors has seen massive amounts of multi-sensor data captured at the individual level.

A dataset is said to be *multi-view* if it comprises multiple data representations—called *views*—and is said to be parallel if objects in the dataset are represented across the multiple views. It may be noted that some objects may not have a representation in certain views. Social media posts comprising [userid, text, image, geolocation] tuples are thus a parallel 4-view dataset, since each social media post would be associated with a userid, some text, (optionally) an image, and a geo-location (if enabled). Multi-view parallel data is increasingly becoming ubiquitous with information being captured using a variety of different modalities, and their prevalence cannot be overemphasized. To outline two exemplary domains, observe that healthcare systems often capture the same disease condition using different medical sensors (e.g., EEG, fMRI, and PET are different ways of capturing neurological information), and criminal records often represent the same crime using modalities such as textual narratives, CCTV footages, audio tapes, and photographs. Since our focus in this paper is multi-view parallel data, we will simply refer to it as multi-view data in the remainder of the chapter.

With the increasing availability of multi-view data across a variety of scenarios, and manual labelling being expensive and impractical in many big data scenarios, multi-view unsupervised learning, the discipline that addresses classical unsupervised learning tasks—viz., clustering [22], dimensionality reduction (e.g., [37]), and outlier detection [18]—over multi-view data, has witnessed massive attention from the scholarly community. In this paper, we provide an overview of the major lines of research in clustering for multi-view data, often referred to simply as *multi-view clustering* (MVC).

Clustering Clustering is a fundamental task in machine learning, and focuses on grouping objects in a dataset into multiple groups, called clusters. The typical criterion for grouping is that objects that are put into the same group should be more similar to each other than objects that are put into different groups. A classical algorithm for clustering [33], the framework of which still forms the backbone of many modern clustering algorithms, dates back to the 1960s. With the decision about the cluster membership of each data object being dependent on the clustering assignments for the other objects in the dataset, the task of clustering often boils down to optimizing for a dataset-wide objective function. A core building block leveraged by clustering algorithms is the choice of the measure of similarity between objects in the dataset being subject to clustering. With different domains and data types from them often requiring similarity measures that are tailored to their needs,

a variety of clustering algorithms have emerged over the last many decades, many of which have become the most highly cited papers in their respective realms; these include clustering algorithms for gene data [15] and time series [29]. The similarity between objects is typically computed as an aggregate of the pairwise similarities of their attributes. In the case of multi-view data, with each data object having multiple views, and each view having a different set of features, there is another level of hierarchy that multi-view clustering techniques need to be cognizant of, to exploit.

Overview of the Paper This paper presents an overview of the state-of-the-art in multi-view clustering, with a focus on covering the different families of methods that have been proposed for the task. Our intent, in slight contrast to a regular survey, is to provide a high-level picture of multi-view clustering methods which is accessible to a generalist who may not be necessarily familiar with the various families of mathematical building blocks that are employed within each method. In addition, we pay particular attention to providing information to enable the reader appreciate the unique characteristics of each family or method, which may make it the preferred method for a particular niche scenario in multi-view clustering. With the focus being on formulations, we omit details about empirical evaluation of the methods, details of which may be obtained from the individual papers. In addition to researchers, we expect our paper to be useful to practitioners who may be looking to decide on a particular multi-view clustering method for usage within their data setting.

Outline We start with outlining the task of multi-view clustering and introducing necessary notation that would be used in the subsequent narrative. Next, we provide a broad outline of the different families of methods that we will cover in our survey, followed by a section each for each family of methods. This will be followed by a section describing a few relaxations to the multi-view clustering problem that have been explored in the literature. We then list a set of datasets that have been explored for evaluating MVC methods and then conclude the chapter.

2.2 Multi-View Clustering: The Task

The input to the clustering task is a multi-view dataset, which we will represent as $X = \{\ldots, x, \ldots\}$. We use $V = \{\ldots, v, \ldots\}$ to represent the set of views in the dataset. Each of these views may comprise multiple attributes, and each multi-view object x takes a value for each attribute within each view, with $x_v.a$ denoting the value it takes for attribute a within view v. In many clustering formulations, the number of desired output clusters is also an input parameter, which we will refer to as k. As alluded to earlier, clustering algorithms typically make use of a quantification of similarity between values for each attribute, which may be rolled up to the object–pair level, both of which are denoted by $s(.,.)$, what is denoted being easily identifiable from the context. Some clustering formulations use a distance function instead of a similarity function, in which case the distance function is referred to as $d(.,.)$.

Table 2.1 Notations

X	A multi-view dataset
x	A multi-view object within X
n	Number of objects in X
V	A set of views represented in a multi-view dataset
v	A view within V
m	Number of views in V
X_v	The subset of X corresponding to view v
x_v	The view v representation of object x
A_v	The attributes or features within view v
a	An attribute within a view
$x_v.a$	The value taken for attribute a within view v by object x
$d(.,.)$	A pre-specified distance function that quantifies the distance between its two input values; we overload this notation to denote both the distance between two values, and its aggregation to the object level
$s(.,.)$	A pre-specified similarity function analogous to $d(.,.)$
$\mathcal{C}$	The set of clusters in the generated clustering
k	The number of clusters in $\mathcal{C}$
C	A cluster within $\mathcal{C}$
$x.C$	The cluster assigned to the object x
$C.p$	The centroid or prototype object for cluster C

Most formulations of MVC output a crisp grouping of all objects to clusters; typical clustering outputs are a partition of the dataset, in the sense each object x is necessarily assigned to a unique cluster, denoted by $x.C$. We use $\mathcal{C} = \{\ldots, C, \ldots\}$ to denote the clustering output by the MVC method. These notations are summarized in Table 2.1.

Thus, at a task level, the MVC task may be seen as accomplishing the following task of using a dataset to arrive at a clustering:

$$X \rightarrow \mathcal{C} \tag{2.1}$$

This may alternatively be written as:

$$\{\ldots, x, \ldots\} \rightarrow \{\ldots, x.C, \ldots\} \tag{2.2}$$

While most MVC methods agree to this general framework and may be described using the above notation which are also tabulated in Table 2.1, there are clustering algorithms that require more terminology to describe. We will introduce such

specific terminology as and when we describe those methods. Additionally, it may be noted that some single-view clustering formulations that may leave out some dataset objects from the clustering output (e.g., [1]) have not been explored for devising MVC methods.

2.3 Overview of Multi-View Clustering Methods

We now group MVC methods into groups based on the technical character of the algorithms. These groupings, while being broadly based on the framework they use, are not fully objective. Some families may legitimately be seen as overlapping; however, we believe that our groupings will help provide a framework towards better understanding the variety and diversity of methods used for MVC. Our groups are listed below, with each group being described in detail in a separate subsequent section.

- **K-Means-Based Approaches:** K-Means, a classical algorithm for clustering [33] single-view data, still holds much sway in the clustering community after half-a-century [21]. Thus, the largest set of MVC methods build upon the K-Means framework.
- **Matrix Factorization:** The dataset corresponding to each view may easily be represented as a matrix with data objects corresponding to rows, and attributes corresponding to columns. This representation easily yields to matrix factorization approaches, particularly those from nonnegative matrix factorization [28]. There have been various flavors of the MVC task that have been addressed using matrix factorization methods.
- **Topic Modelling-based Approaches:** Topic models [3] seek to model documents as a mixture of topics, with each word being drawn from a topic that has representation in the document. There have been several methods that draw upon the idea of topic modelling, of which probabilistic latent semantic analysis [19] has seen much uptake towards crafting methods for MVC.
- **Spectral Methods:** In a broad sense, spectral methods make usage of the spectrum, i.e., the set of eigenvalues, and of the similarity matrix of the data to perform clustering of the dataset. These, at a fundamental level, relate to graph representations that model the similarities between data points. Spectral clustering methods, such as [38], have seen much interest in the image processing community, and have been adapted to the general MVC task as well.
- **Techniques using Exemplars:** Exemplars are typically used to refer to a proxy object, whether it be for a cluster or an individual data object. These lend well to belief and affinity propagation models (e.g., [17]), which have also inspired the design of some MVC methods.
- **Miscellaneous:** In this family, we cover MVC methods that do not necessarily fit well within any of the above classifications. These include techniques that are inspired by canonical correlation analysis [40] and co-clustering [12].

The expert would rightly observe that there is a nontrivial overlap between families; for example, K-Means clustering may be seen as an instance of matrix factorization, and so could be topic modelling. Thus, our categorization is intended to give an organization that a researcher would easily relate to, and does not imply that the separate families are disjoint or unrelated.

2.4 K-Means Variants for MVC

We first start with a description of the K-Means clustering method [33], a popular clustering algorithm for single-view data. K-Means targets to produce a pre-specified number of clusters, denoted as k, in the output. Each cluster is represented by a prototype, a virtual data object that is modelled as the mean of all the data objects assigned to the cluster. Simplistically, K-Means creates a cluster assignment towards optimizing the following objective function:

$$\sum_{x \in X} d(x, x.C.p) \tag{2.3}$$

Thus, the clustering allocation is made in a way that the sum of distances of each object x to the prototype of cluster to which it is assigned, denoted by $x.C.p$, is minimized. The distance function is typically modelled as the L^2 norm[1] of the vector of distances between the objects over the set of attributes under consideration.

$$d(x, y) = \sum_{a \in A} (x.a - y.a)^2 \tag{2.4}$$

In the single-view formulation, there is only one set of attributes, A, given that we only have a single-view representation for each object. The K-Means formulation may be thought of as an instance of the Expectation Maximization algorithm [11], where two sets of parameters, the cluster assignments $\{\ldots, x.C, \ldots\}$, and the cluster centroids $\{\ldots, C.p, \ldots\}$ are optimized in an alternating fashion iteratively until convergence. As may be obvious for a reader familiar with EM, the cluster assignment corresponds to the E-step and the centroid learning corresponds to the M-step. Since K-Means could converge at local minima, the initialization of clusters in order to kick-start the iterative learning process is often regarded to be critical.

[1] http://mathworld.wolfram.com/L2-Norm.html.

2.4.1 Alternating K-Means for Two-View Data

In probably what could be among the earliest works in MVC [2], extensions to the K-Means method for multi-view data were proposed. This was specifically tailored to two-view data, with a focus on document and webpage clustering. In what could be regarded as a simplistic extension to K-Means, they propose to interleave the EM steps corresponding to each view. Starting from an initialized cluster membership, a sequence of M and E steps are performed by using just the data from one view (to re-emphasize, the data from the other view is not used) to arrive at a cluster allocation for the data objects. The cluster allocation is taken to then perform the M and E steps using just the data from the *other* view. In summary, each iteration involving the sequence of M and E steps uses one particular view, with the immediate next iteration shifting the focus to the other view. Thus, the clustering information across views flows across iterations through the clustering allocations. At the end of a sequence of such view-alternating iterations, two sets of clustering allocations are produced, each one corresponding to the latest allocation from each view. These clustering allocations are then merged in a post-processing step in order to arrive at a single clustering for the data objects.

At the task level, it is notable that the method is proposed with two-view data in mind; however, a simple extension that executes iterations in round-robin fashion by cycling through three or more views may be envisaged for multi-view data incorporating three or more views (the empirical performance would need to be investigated). Another notable feature of this method is that there is no provision for weighting the two views differently with regard to their influence in the final clustering output. Such differential weighting, we will see, has been the focus of many later K-Means variants.

2.4.2 Max/Min Fusion Within K-Means

A very trivial extension to the K-Means objective function can be arrived at by just summing up the distances across views, and optimizing for the sum of distances:

$$\sum_{x \in X} \sum_{v \in V} d(x_v, (x.C.p)_v) \tag{2.5}$$

where $(x.C.p)_v$ denotes the view v representation of the cluster centroid to which x belongs. The view-specific distance function being simply the L^2 distance over the attributes in that view.

$$d(x_v, y_v) = \sum_{a \in A_v} (x_v.a - y_v.a)^2 \tag{2.6}$$

It may be noted that the sum of distances is clearly equivalent to the average of distances across views, with the number of views being the same across all data objects. Motivated by scenarios from community question answering systems, where the multi-view dataset comprises two-view objects with question (Q) and answer (A) text forming separate views, Deepak [9] proposes replacing the average/sum aggregation of distances across views by the min function. Thus, the membership of an object in a cluster is determined by computing the distance of the object to each cluster prototype aggregated across views, the aggregation performed by using the min function. Thus, the objective function changes to:

$$\sum_{x \in X} \min\{d(x_v, (x.C.p)_v) | v \in V\} \tag{2.7}$$

In summary, if x is very proximal to a cluster prototype in one of the views, it will be assigned to that cluster regardless of the distance of x to the same cluster's prototype in other views. This is motivated by scenarios where the similarity information is localized in certain views; for example, this formulation places QA pairs that are highly similar on *either* the Q or A views, within the same cluster.

With the min function being non-differentiable, Deepak [9] proposes usage of a differentiable approximation involving exponentiation. The approximation additionally is applicable to using max aggregation instead of min, though the focus of their work is min aggregation. The paper observes that this formulation, much like the previous one, is trivially extensible to more than two views. In another point of similarity with the earlier work, there is no intrinsic method to pre-specify that one of the views should influence the clustering decisions more. However, the paper notes that scaling $d(x_v, y_v)$ by a view-specific weight would allow the user to make such tuning; in such a case, a lower weight would lead to the view being able to influence the clustering more, the aggregation function being min.

2.4.3 View-Weighted K-Means with $L^{2,1}$ Norm

Let us now look at the relation between K-Means and matrix factorization. Single-view (relaxed [13]) K-Means clustering may be written as a nonnegative matrix factorization problem with the objective:

$$\min_{G,F} ||X - GF||_2^F \tag{2.8}$$

$$s.t\ G_{ij} \in \{0, 1\}, \sum_{j=1}^{k} G_{ij} = 1$$

where X is the input single-view data matrix (given that single-view is a specialization of multi-view with the number of views being unity, we use the same variable X) with n rows and as many columns as there are attributes in the view, G being a clustering indicator matrix of $n \times k$, and F being a cluster centroid matrix with one row per cluster. It may be noted that the constraints placed on G enforce that each object is assigned to only one of the k clusters, with k being a user-specified parameter. The solutions, G (clustering assignment) and F (cluster centroids) are arrived at by minimizing the square of the Frobenius norm[2] of the difference matrix (indicated by $||M||_F^2$), which is essentially the sum of the squares of all elements in the difference between X and GF matrices.

In extending this to multi-view data, we would need to account for m matrices of data, one corresponding to each data view, as well as m cluster centroid matrices (again, one for each view). However, given the MVC task, there needs to be a single cluster indicator matrix across all views. This would yield

$$\min_{G,\{...,F_v,...\}} \sum_{v \in V} ||X_v - GF_v||_2^F \tag{2.9}$$

with the usual constraints on the G matrix being applied. In contrast to such an extension, Cai et al. [7] allow for view weights and use the $L^{2,1}$ norm instead of the Frobenius norm, leading to the following:

$$\min_{G,\{...,F_v,...\}} \sum_{v \in V} (w_v)^{\lambda} ||X_v - GF_v||_{2,1} \tag{2.10}$$

For a matrix M, the $L^{2,1}$ norm sums up, across all column vectors of M, the sum of squares of their components. In the above case, it comes down to the L^1 norm in the data-points direction, and the L^2 norm in the features direction. The $L^{2,1}$ norm is popular in scenarios where robustness is desired (e.g., [36]), a feature that the authors of [7] argue as being important in MVC. The w_v are view weights that are learnt within the optimization process, with λ being a parameter that would control the weights distribution. The modified objective leads to different update rules that are detailed in [7]. Unlike the earlier two papers, this method quantifies the influence each view could have in the clustering process; however, the views are learnt in the process of optimization (*not* pre-specified by the user).

2.4.4 View and Attribute Weighting Within K-Means

After work on view-specific weighting, almost as a natural next step, algorithms for MVC were proposed that use attribute-weighting within views. Jiang et al. [24]

[2] http://mathworld.wolfram.com/FrobeniusNorm.html.

propose extending the basic K-Means model along that direction, leading to an objective function:

$$\sum_{x \in X} \sum_{v \in V} (w_v)^{\alpha} \sum_{a \in A_v} (z_a)^{\beta} \big(x_v.a - (x.C.p)_v.a\big)^2 \tag{2.11}$$

$$\sum_{v \in V} w_v = 1$$

$$\forall\, v \in V, \sum_{a \in A_v} z_a = 1$$

Thus, the distance between the data point and the cluster prototype along each attribute within each view is scaled twice, first by the view-specific weight for the view, and second by the attribute-specific weight. Additionally, the weight distributions are controlled by respective exponents α and β. Further, as indicated above, it is enforced that the view weights across views sum to unity, as well as that the attribute weights across attributes within *each* view sum to unity. The learning process sequentially learns: (1) cluster centroids, (2) attribute weights, (3) view weights, and (4) cluster assignments, in four different steps within each iteration.

Along a similar direction, Chen et al. [8] propose the usage of additional terms quantifying the negative entropies of the view and feature weights, so most views and features are called into play, unless there is a compelling reason to focus on just a few. Their objective function assumes the following form:

$$\sum_{x \in X} \sum_{v \in V} w_v \sum_{a \in A_v} (z_a) \big(x_v.a - (x.C.p)_v.a\big)^2 + \eta \sum_{a \in \left(\cup_{v \in V} A_v\right)} z_a \log(z_a)$$

$$+\lambda \sum_{v \in V} w_v \log(w_v) \tag{2.12}$$

along with the sum-to-unity constraints on z and w as earlier. Including the negative entropies in minimization is motivated by the previous work on similar lines [25]. Similar to what is done in [24], the four sets of parameters are sequentially optimized for within each iteration.

In yet another variation, Xu et al. [47] propose the usage of a regularizer to control the sparsity over the feature weights, leading to the following objective:

$$\sum_{x \in X} \sum_{v \in V} (w_v)^{\alpha} \sum_{a \in A_v} (z_a) \big(x_v.a - (x.C.p)_v.a\big)^2 + \beta \sum_{v \in V} ||\{z_a | a \in A_v\}||^2 \tag{2.13}$$

along with the sum-to-unity constraints as earlier. The regularizer avoids attaining a configuration where only a few features are selected, which would lead to a small meaningless (sic) objective value despite being small.

All the methods covered in this subsection, as in the previous, learn view and attribute weights within the learning process. This leads them to quantifying the influence of attributes and views; however, being part of the learning process, they are not pre-specifications from the user side on their respective influences.

2.5 Matrix Factorization Approaches to MVC

Much like in the previous section, we start with outlining the basic framework of nonnegative matrix factorization (NMF), the matrix factorization family that has been explored widely in clustering. When X, the data matrix (each row being a data object), is nonnegative, NMF seeks to arrive at a decomposition of it into two matrices using, in most cases, the following objective function:

$$\min_{G,F} ||X - GF||_F^2 \tag{2.14}$$

where G is an $n \times l$ matrix, and F is an $l \times |A|$ matrix, and each of their elements are constrained to be nonnegative. We observed earlier that when $l = k$ and one-of-k coding constraints are imposed on G, it comes closer to a K-Means clustering formulation. In a sense, for general l, one could consider the F matrix as modelling l object prototypes, with each object in X being constructed as a linear combination of the prototypes in F using the weights from G. Under this model, F may be called as the *basis matrix* and G is the *coefficient matrix*. The coefficient matrix may be considered as providing a representation for each object in X within a latent (low-rank) space. Further, a simple clustering may be achieved by associating each data object in X with one of l clusters, specifically, the cluster with which it has the highest coefficient. We will consider three MVC methods that build upon NMF, in this section.

2.5.1 Joint NMF for MVC

In what is probably the first method using NMF directly for MVC, Liu et al. [30] propose factorizing the different data matrices, i.e., X_vs, for MVC. However, since a single clustering solution across the views is what is desired, the separate factorizations need to be done jointly in order to achieve *similar* coefficient matrices from the different factorizations. Further, the optimization function also involves learning of a cross-view coefficient matrix that will eventually be used in order to generate the clustering. The proposed objective function is thus the following:

$$\min_{G_v, F_v, G^*} \sum_{v \in V} ||X_v - G_v F_v||_F^2 + \sum_{v \in V} w_v ||G_v - G^\circ||_F^2 \tag{2.15}$$

Note that instead of doing pairwise comparisons of view-specific coefficient matrices, the above form has a term for a cross-view consensus matrix G° from which the deviations are quantified in the second term. In order to ensure that the different coefficient matrices are comparable, they additionally impose that the basis vectors (i.e., within F) have components summing up to unity. Notationally, the constraint is the following:

$$\forall 1 \leq i \leq k, \left(\sum_{j=1}^{l} F_{ij} = 1 \right)$$

Do note that this is in addition to the nonnegativity constraints on all factorized matrices. The optimization is performed in an iterative framework, where, within every iteration, the F_v and G_v for all views are optimized for, in addition to optimizing for G°. At the end of the learning process, G° is achieved, which may be used as a new representation for objects in X to be subjected to single-view K-Means to arrive at a clustering. Alternatively, for each object, the cluster with which it has the highest coefficient (only if $l = k$) may be assigned as the cluster to which the object belongs, resulting in an MVC output.

Unlike the methods seen so far, the set of view weights, w_vs, are pre-specified weights that are not altered/learnt in the course of the optimization framework.

2.5.2 Manifold Regularized NMF for MVC

The joint NMF was closely followed by another extension of NMF for MVC that is based on manifold regularization [51]. They draw inspiration from previous work [6] that remedies a "deficiency" in classical NMF, one that relates to preserving space geometry. Informally, they argue that two data objects in a dataset being close enough in the intrinsic geometry of the distribution should entail that their representations (i.e., the coefficient representations from G) be close to each other. In particular, when a graph representation of data objects is available, objects that are connected to each other need to be proximal in their NMF representations as well. Consider a graph representation of objects in X where an edge is induced between pairs of objects if one of them appears in the other's k nearest neighbors, and let L be the $n \times n$ Laplacian matrix[3] of such a graph. Cai et al. [6] propose that preservation of intrinsic geometry across objects in X is better achieved if the following regularizer be added to the usual NMF objective:

$$\cdots + tr(G^T L G) \tag{2.16}$$

[3] https://en.wikipedia.org/wiki/Laplacian_matrix.

where $tr(X)$ denotes the trace[4] of the matrix X. With multiple views entailing multiple Laplacian matrices (each view-specific matrix represented as L_v), Zong et al. [51] model the task using the following objective:

$$\min_{G_v, F_v, L^\circ, G^\circ} \sum_{v \in V} D(X_v || G_v F_v) + D(G_v || G^\circ) + D(L_v || L^\circ) + \lambda \, tr((G^\circ)^T L^\circ G^\circ) \tag{2.17}$$

where $D(X||Y)$ denotes the cost function to quantify the difference between X and Y. While the first and second terms are familiar from having encountered in [30] above, the third term forces the learning of a cross-view consensus graph structure and the fourth term incentivizes preservation of local geometry in the consensus space. The paper proposes different variations of multimanifold regularized NMF based on the above framework, each of which vary on some aspects of the objective function construction (some variants also provision for view weights) and the computation involved in the optimization process.

2.5.3 Deep Matrix Factorization for MVC

Semi-NMF An attractive feature of NMF is the interpretability with each data point being represented as a linear combination of the different bases (from the basis matrix) with nonnegative coefficients. Semi-NMF [14] is a modification of NMF that relaxes the nonnegativity constraints on the data and the basis matrix, while retaining interpretability by enforcing that the coefficient matrix is nonnegative. Informally, it makes NMF applicable for mixed-sign data (i.e., some entries of the X matrix may be negative) and drops the nonnegativity constraint on the entries in the F matrix.

Extending Semi-NMF for MVC Zhao et al. [50] build upon Semi-NMF to formulate a sequence of transformations via many basis matrices. For single-view data, this assumes the form:

$$\min_{G, \forall i, F_i} ||X - G F_r F_{r-1} \ldots F_1||_F^2 \tag{2.18}$$

where there are r basis matrices to be estimated, their dimensionality controlled appropriate to the complexity of the model desired. For multi-view data, they enforce that the representation is common across all views (i.e., G is shared) and add the manifold regularization term in order to ensure that the common representation preserves the local geometry within each view. This leads to the following objective function construction:

[4] https://en.wikipedia.org/wiki/Trace_(linear_algebra).

$$\min_{G,\forall v,w_v,\forall i,v,F_i^v} \sum_{v\in V} (w_v)^{\alpha} \left(||X_v - GF_r^v F_{r-1}^v \dots F_1^v||_F^2 + \beta tr(G^T L_v G) \right) \tag{2.19}$$

with L_v, as earlier, being the graph Laplacian for view v constructed using k nearest neighbor-induced edges. The usual sum-to-unity constraints are applied on the view weights, w_v. The learnt representation G is then clustered to arrive at an MVC result. The length of the sequence of transformations, it may be noted, much like the number of layers in a neural network, may be controlled to learn representations at the level of data features at different levels of abstraction; r may thus be set appropriately. It is also noteworthy here that unlike the earlier NMF approaches, this one can handle mixed-sign data.

2.6 Topic Modelling-Based Approaches

Topic-modelling, a technology from the text processing community, considers learning a layer in between documents and their component words, called a set of topics. Each topic is usually represented as a probability distribution over words in the vocabulary, with each topic having a higher probability associated with words that typify the topic. Using the observed documents and the words they comprise, a set of topics, their number controlled by a pre-specified parameter, is learnt. We provide a brief overview of probabilistic latest semantic analysis [19], a popular model for learning topics.

A topic is modelled, as outlined earlier, as a probability distribution over the vocabulary of words, and a document has a probability distribution over the topics. Let these be represented as $P(w|t)$ and $P(t|d)$, respectively. Then, the likelihood of occurrence of a word w in document d may be written out as:

$$P(w,d) = \sum_{t\in T} P(w|t)P(t|d) \tag{2.20}$$

where T is the set of topics. Now, given a corpus of documents, topics may be learnt by maximizing the likelihood:

$$\max_{\forall w,t,P(w|t),\forall t,d,P(t|d)} \prod_{d\in D} \prod_{w\in W} \left(\sum_{t\in T} P(w|t)P(t|d) \right)^{n(d,w)} \tag{2.21}$$

with $n(d,w)$ indicating the number of times word w has appeared in document d. The log-likelihood assumes the following form:

$$\mathcal{L} = \max_{\forall w,t,P(w|t),\forall t,d,P(t|d)} \sum_{d\in D} \sum_{w\in W} n(d,w) \times \log \left(\sum_{t\in T} P(w|t)P(t|d) \right) \tag{2.22}$$

The document–topic probabilities and topic–word probabilities are typically learnt alternatively in the E and M steps of an Expectation Maximization framework. Though the above framework is designed for document datasets, it may be trivially seen that it applies to any dataset if words are considered as attributes and $n(d, w)$ is interpreted as the weight associated with the feature w in data object d.

2.6.1 Collaborative PLSA for Clustering Two-View Data

Upon estimation of the topics, one could view the topics as clusters. Similar to the NMF case, a trivial clustering is achieved with each document (i.e., data object) assigned to the topic with which it has the highest affinity. Alternatively, the $P(.|d)$ vectors may be treated as unit L^1 length representations, which could be subject to a clustering method such as K-Means.

In [23], an extension of PLSA to two-view data is provided, with applications to clustering. The idea is that the $P(.|d)$ vectors derived from considering each of the dataset views separately could be different, but a middleground could be found by adding a soft constraint in order to bring the vectors together. This results in the following objective function:

$$\max_{\forall w,t,v_i P_{v_i}(w|t),\forall t,d,v_i P_{v_i}(t|d)} w_{v_1}\mathcal{L}_{v_1} + w_{v_2}\mathcal{L}_{v_2} - \beta \sum_d \sum_t (P_{v_1}(t|d) - P_{v_2}(t|d))^2 \tag{2.23}$$

Here, $P_{v_i}(.|.)$ denotes the estimates arrived at for view v_i. Thus, the objective function is simply the weighted sum of the log-likelihoods for the two views separately (denoted by $\mathcal{L}_{v_i}$) discounted by the square of the L^2 distance between the $P(.|d)$ vectors from either view, summed up over all documents. The view weights w_v are pre-specified and are expected to be part of the user input. The last term forces the optimization to learn two separate document representations from separate views, but forces that the representations are close together. The cluster assignment for each document is arrived at using the max aggregation.

$$d.C = \arg\max_t \{P_{v_1}(t|d),\ P_{v_2}(t|d)\} \tag{2.24}$$

The method, thought built upon topic modelling, has also been evaluated over image datasets; refer [23] for details.

2.6.2 Voting-Based MVC

In another method [26] that builds upon PLSA topic modelling, a two-phase approach is proposed. In the first phase, the separate-view datasets are subjected

to separate PLSA topic modelling to arrive at separate view-specific topics; the number of topics are kept the same across the views. This is followed by a second phase, where documents that share similar $P_v(.|d)$ vectors are pre-assigned to some groups, these groups eventually feeding into the final clustering. The documents that cannot be assigned to the groups are represented using a concatenated representation derived by simply collating the view-specific representations. All the documents (grouped as well as ungrouped) are then subjected to another round of topic modelling, this time a cross-view one. Within the iterations, it is enforced that the pre-assigned documents do not switch groups, whereas the other documents are free to switch groups. This leads to a final cross-view $P(.|d)$ vector, which is then translated to a clustering.

2.7 Spectral Methods

Spectral Clustering (e.g., [35]) operates on a graph formed by the data objects as nodes and the (usually weighted) edges representing similarities between data points. As we saw earlier in the section on matrix factorization, a graph can be induced from a dataset X by forming edges between objects if one of them figures in the other's k nearest neighbors; there could be other intuitive ways of translating a relational dataset into a graph. Spectral clustering algorithms are favored in cases where the desired clustering output involves irregular shapes. A simple spectral clustering algorithm over a graph represented as a similarity matrix S may be outlined as below:

- **Graph Laplacian:** Form the Laplacian matrix of the graph from the $n \times n$ similarity matrix in the input, which we will denote as $\mathcal{L}$. This would be an $n \times n$ matrix.
- **Eigen Decomposition:** Perform eigen decomposition over $\mathcal{L}$ and choose the top-k eigenvectors. Let the eigen vectors be stacked column-wise to form an $n \times k$ matrix U.
- **Normalization:** The rows of the matrix U are normalized to form a normalized matrix $\hat{U}$. Each row may now be treated as a representation of the corresponding data point.
- **Clustering:** The normalized vectors can now be subjected to a conventional clustering algorithm such as K-Means to arrive at a clustering.

In a fully connected graph, spectral clustering methods may be thought of as solving a version of the min-cut problem, i.e., identifying a set of k connected components by discarding a few edges. It is also notable here that the k eigen vectors of the Laplacian matrix may be thought of as signatures of the k clusters in the output. We will now consider MVC methods that build upon ideas from spectral clustering.

2.7.1 Co-training Spectral Clustering Over Two-View Data

Co-training (e.g., [4]) proposes searching in two different hypothesis spaces (e.g., two different clusterings in two different data views, in the case of MVC) while gradually swaying them towards each other, so that the eventual choices of hypotheses agree reasonably well with each other.

Kumar and Daumé [27] blend co-training with spectral clustering in order to devise an MVC solution for two-view data. The spectral clustering is solved in the individual views separately, in order to arrive at two top-k eigen vector column-stacked matrices U_1 and U_2. This is followed by modifying the similarity matrix S_1 using information from U_2 (i.e., the clustering exposited by the eigen vectors in the second view) and modifying the similarity matrix S_2 using information from U_1. The modification is performed by projecting the columns of S_1 (S_2) on the eigen vectors from U_2 (U_1) and re-projecting them back to the n-dimensional space. The projection operation using S_1 implicitly discards some information that is not in line with the clustering information from U_2, and vice versa. The modified S_1 (S_2) is then used in the next iteration for deriving U_1 (U_2). Each projection–re-projection operation nudges the similarity matrix from one view towards the clustering structure from the other view. Over the course of many iterations, the clusterings from either view converge, leading to an MVC solution.

While extensive empirical results are presented for two-view data, the authors also outline that the framework may be extended for data with more than two views by performing the projection–re-projection operation using the aggregated eigen vectors from all views (but for the one whose similarity matrix is being modified).

2.7.2 Pareto-Optimal Spectral MVC

The co-training approach that we just saw makes an implicit assumption that the two views are compatible and agree on a single clustering of the data. However, there may be cases where there are disagreements between data views. A relaxation of the agreement assumption is used in order to devise another spectral MVC method [42]. Consider two data views v_1 and v_2 as earlier. Now, given the graphs induced by the respective view-specific similarity matrices, each cut c of the graph (note that the graph nodes are common across views, being the data points in parallel data) that produces k connected components, may be costed for each view; the cost, as may be intuitive, quantifies how much the view *dislikes* the cut. Let $\mathcal{C}(v, c)$ denote the cost of the cut c over the view v. Now, a space of 2-d vectors may be defined as follows:

$$\mathcal{S} = \{(\mathcal{C}(v_1, c), \mathcal{C}(v_2, c) | c \in \Theta\} \tag{2.25}$$

where Θ is the set of all nontrivial cuts over graphs across the two views. The optimal cut for the first view, i.e., $\arg\min_{c \in \Theta} \mathcal{C}(v_1, c)$, may differ from the optimal

cut for the second view when there are disagreements across the data views. Thus, Wang et al. [42] propose searching for the set of pareto-optimal cuts, which is defined as the skyline [5] of $\mathcal{S}$. Towards defining the skyline, it is useful to understand the domination operator. A vector (x, y) is said to dominate another (x', y') if and only if the following conditions are satisfied:

$$x \leq x' \wedge y \leq y' \wedge (x \neq x' \vee y \neq y') \tag{2.26}$$

Thus, (x', y') is dominated as long as it is at least as far as (x, y) from the origin on both axes, as long as both the vectors are not equal. The skyline of $\mathcal{S}$ are the set of vectors that remain when all dominated vectors are filtered out. When vectors in the cost-space $\mathcal{S}$ are considered, it amounts to finding vectors corresponding to a set of cuts such that it is impossible to find another cut that has lower cost on both the data views.

The set of pareto-optimal cuts could be very large, especially if the disagreements between the data views are high. Each pareto-optimal cut yields a clustering of the multi-view dataset. However, if one were interested in aggregating the set of pareto-optimal cuts to form a single MVC result, the cluster indicator vectors corresponding to each pareto-optimal cut could be subject to a further round of clustering.

The proposed method easily generalizes to more than two data views, with V view datasets entailing a skyline search over V length vectors. It may, however, be noted that the size of the pareto-optimal set usually increases quite fast with the number of dimensions of the vectors involved.

2.8 Exemplar-Based Approaches

Exemplar-based approaches have been popular for clustering (e.g., [17]), with the clustering process operationalized through message passing. Exemplars are a small set of data points chosen from the dataset, each of which stand for a cluster in the output. Each data point is associated with one of the data points in the set of exemplars, and that association determines the cluster membership as well. The clustering process is itself initialized by setting each data point as its own exemplar, and gradually shrinking the set of exemplars through associating each data point with a different exemplar typically based on both: (1) how similar the data point is, to the exemplar, and (2) how much other data points also prefer being associated with the exemplar. The latter factor ensures that the set of exemplars are shrunk progressively in order to achieve a small set of clusters in the eventual output. Prior information about suitabilities of particular data points to be exemplars is easily incorporated by weighting the second criterion highly for such data points. Exemplar-based clustering is attractive in that there is an automatic choice of prototypical data object for each cluster, one that could act as a summary in scenarios where human perusal of clusters is necessary. We describe two approaches that generalize from exemplar-based clustering for MVC.

2.8.1 Simple Similarity Matrix Aggregation

Scientific journal papers may be treated as multi-view data comprising the text view, and the citation profile. An exemplar-clustering approach designed for clustering such scientific datasets, proposed in [34], simply takes the $n \times n$ similarity matrix from the separate views and forms an aggregated similarity matrix using a weighted-sum form:

$$\mathcal{S} = \alpha \mathcal{S}_T + (1 - \alpha)\mathcal{S}_C \tag{2.27}$$

where $\mathcal{S}_T$ and $\mathcal{S}_C$ are the similarity matrices from the text and citation views, respectively, and α controls the relative weighting between the two views. The aggregated similarity matrix $\mathcal{S}$ is then subject to exemplar-based affinity propagation to arrive at an MVC result.

2.8.2 Affinity Propagation with Cross-View Agreement

A more recent work [44] introduces a more sophisticated approach towards exemplar-based clustering through incorporating an explicit criterion to enforce an agreement between the clusterings from across views. We describe the objective function they optimize for, leaving the interested reader to refer to the paper for finer details of the message passing framework. Within a single view dataset, X, the objective function for an exemplar approach may be written down as:

$$\mathcal{J} = \sum_{x \in X} \mathcal{S}(x, c_x) + \sum_{x \in X} \begin{cases} -\infty & x \neq c_x \wedge (\exists x' \in X, x = c_{x'}) \\ 0 & \text{otherwise} \end{cases} \tag{2.28}$$

where c_x denotes the exemplar of the data object x. The second term effectively forces a data point that is an exemplar for a different data point, to be an exemplar for itself as well, an intuitively motivated condition. The first term quantifies the similarity between an object and its exemplar. Estimating the $\{\ldots, c_x, \ldots\}$ by maximizing $\mathcal{J}$ would intuitively lead to a single-view clustering solution for X.

The MVC work [44] extends this framework for exemplar-based clustering by adding all the view-specific $\mathcal{J}$s along with a pairwise cross-view agreement term as follows:

$$\alpha \sum_{v \in V} \mathcal{J}_v + (1 - \alpha) \sum_{v \in V} \sum_{v' \in V} I(v \neq v') \sum_x \frac{|\mathcal{N}_k^v(x_c^v) \cap \mathcal{N}_k^{v'}(x_c^{v'})|}{|\mathcal{N}_k^v(x_c^v) \cup \mathcal{N}_k^{v'}(x_c^{v'})|} \tag{2.29}$$

where $\mathcal{N}_k^v(x_c^v)$ denotes the set of k nearest neighbors to the exemplar of the data point x within view v (note that each data point has a different exemplar within

each view), when neighbors are computed based on the similarity matrix from the view v. Thus, the last term is a Jaccard similarity term between the pairs of neighborhoods across two different views, for view-specific exemplars associated with the same data object. Estimating the view-specific exemplars for each data object by maximizing this objective leads to a different clustering solution for each view; however, they are expected to be in reasonable agreement, due to having considered the cross-view neighborhood agreement in the optimization. The view-specific clusterings may then be aggregated to arrive at a single MVC result.

2.9 Other Approaches to MVC

Having considered the major families of techniques for MVC, we now turn our attention to some methods that cannot be easily categorized into any of the previously discussed categories. We attempt to provide an illustrative sample of the variety of approaches to MVC, over and above those covered earlier.

2.9.1 Multi-View Ensemble Clustering

Ensemble Clustering encompasses the family of clustering methods (e.g., [16]) that can fuse multiple clusterings into a single consensus clustering. There has been recent work on spectral ensemble clustering [31] that suggests that multiple clusterings of the same set of data X may be leveraged to form a co-association matrix as follows:

$$\mathcal{S}_{xx'} = \frac{|\{C|C \in \mathcal{C} \wedge x.C = x'.C\}|}{|\{C|C \in \mathcal{C}\}|} \tag{2.30}$$

Informally, $\mathcal{S}_{xx'}$ captures the fraction of clusterings in $\mathcal{C}$ that puts x and x' in the same cluster. Now, by minimizing the objective that uses the graph Laplacian $L_{\mathcal{S}}$ of $\mathcal{S}$:

$$\min_{H} tr(H^T L_{\mathcal{S}} H) \tag{2.31}$$

with the constraint $H^T H = I$ leads to an estimate of H that can be used to identify a partitioning of data points in X such that the similarities in $\mathcal{S}$ are respected.

The multi-view extension [39] of this ensemble clustering method seeks to learn a low-rank cross-view representation of the data objects as $\mathcal{Z}$ and an associated H representing a cross-view clustering result. Towards ensuring that the low-rank representation $\mathcal{Z}$ is in agreement with the various view-specific similarity matrices $\mathcal{S}_v$, a view-specific constraint is imposed, leading to an iterative optimization

formulation. We refer the interested reader to the paper for details of the full objective function.

2.9.2 Co-clustering for Multi-View Datasets

Co-clustering (e.g., [12]) is a paradigm for clustering words and documents simultaneously in document datasets. Adopting a variant of co-clustering, Hussain and Bashir [20] consider learning document–document (i.e., pairwise) similarities interleaved with learning of word–word similarities within a single-view document dataset. This leads to an iterative algorithm that, when initialized with simple similarities based on word and document co-occurrences, produces a refined estimate of document–document similarities at the end, one that could be used to form a clustering. Given a clustering, the ratio of intra-cluster document similarities to document similarities across the corpus leads to an intrinsic (i.e., unsupervised) goodness measure for the similarity matrix.

Further, Hussain and Bashir [20] propose to extend the single-view method to multi-view datasets by using cross-view learning within each iteration to arrive at an MVC method. In one variant, the better document–document similarity estimates from across views (when evaluated using the intrinsic goodness measure against a clustering) are chosen to feed into the next learning iteration. They provide two additional variants that differ on how the information is fused across views within the iteration.

2.9.3 Multi-View Clustering via Canonical Correlation Analysis

Canonical Correlation Analysis (CCA [40]), which bears relations with principal component analysis for single-view data, is a classical method for identifying directions along which the separate views of a two-view dataset are correlated. In an extension based on CCA, Livescu et al. [32] propose to identify the top-k CCA directions and then project the separate datasets, i.e., X_vs, to those directions. This is then followed by a conventional clustering on the transformed view-specific datasets. Their work is built upon an independent assumption, whereby parallel samples from the same cluster (i.e., two data samples belonging to the same object, but from the separate views) can be regarded as independent given the cluster label. This is intuitive for cases such as multimodal data, e.g., text and video, whereby the dependence between the text and video from the same person may be largely attributed to the identity of the person.

2.10 Variations of the MVC Task

Having considered a variety of methods that address the MVC task, we now look at a few variants of the MVC task that have been explored in the literature.

2.10.1 Multi-View Clustering with Unmapped Data

The conventional setting of MVC, the setting that has occupied all our attention until now, has been the case where the multi-view data is parallel. Thus, the ith row of X_v relates to the same object as the ith row of $X_{v'}$. However, as observed in [49], such comprehensively parallel data is not always available. As an example, in certain cases, the cross-view linking information could be very sparse in that, only information of presence or absence of linkages between a few pairs of data objects may be available. Zhang et al. [49] consider the case where different views could potentially consist of different sets of objects (and even varying number of objects across views), and there are two sets of linkage information available, as follows:

$$\text{ML} = \{(i_v, j_{v'}) | 1 \leq v, v' \leq |V|\} \tag{2.32}$$

$$\text{CL} = \{(i_v, j_{v'}) | 1 \leq v, v' \leq |V|\} \tag{2.33}$$

Informally, ML and CL are sets of pairs of objects, which could be from across views, which are deemed to must-link (i.e., should be part of the same cluster) or cannot-link (analogously, should be in different clusters), respectively. If prior information is available that two objects from across views—specifically, the ith object from view v and the jth object from view v'—are indeed associated with the same underlying entity, the pair corresponding to them—i.e., $(i_v, j_{v'})$—may be then added to the set ML. Analogously, two objects for whom information is available that they are dissimilar, and possibly do not even belong to the same cluster, they may then be added to the CL set.

The work [49] proposes modifying the NMF to account for these constraints as follows:

$$\sum_{v \in V} ||X_v - G_v F_v||_F^2 + \sum_{(i_v, j_{v'}) \in \text{ML}} \sum_{x=1}^{k} (G_v[i][x] - G_{v'}[j][x])^2$$
$$+2 \times \sum_{(i_v, j_{v'}) \in \text{CL}} \sum_{x=1}^{k} G_v[i][x] \times G_{v'}[j][x] \tag{2.34}$$

The first term is the usual NMF loss aggregated across views, whereas the second and third terms relate to the ML and CL constraints, $G_v[i][x]$ indicating the xth

element in the vector corresponding to the ith element from the coefficient matrix (which, as we saw earlier, is the clustering indicator matrix for clustering scenarios). The second term quantifies the *dissimilarity* between the objects involved in ML constraints (that needs to be minimized), and the third term quantifies the *similarity* between objects in CL constraints (that needs to be minimized as well). Each data point is then associated with the cluster with which it has the highest coefficient in the respective G_v matrix, leading to a clustering output.

2.10.2 Multi-Task Multi-View Clustering

Multi-task clustering is the setting where multiple-related tasks are to be addressed together in order to achieve a better effectiveness. For example, clustering web images from Chinese and English websites may be considered as two related tasks. The images trivially come from the same space, that of images; however, the text surrounding the images, which may hold valuable cues to arrive at a clustering solution, come from different spaces, that of Chinese and English words, respectively. Within each task, the relationship between the component views is to be ensured in that the clustering they exposit needs to be consistent. On the other hand, within each view that is shared across tasks, the separate tasks should use the same notion of similarity. Multi-task Multi-view Clustering [48] considers accomplishing the two tasks of multi-task and multi-view learning for the clustering setting, within the same framework. This utilizes the shared data objects across views, and the shared views across tasks, in order to mutually enhance the accuracies of the task-specific results. They compartmentalize the overall problem into three separate considerations as follows:

- **Within-view-task Clustering:** For a combination of a chosen view and chosen task, the solution to the clustering problem may be achieved by a co-clustering method that simultaneously partitions both the data objects and the attributes under consideration. Their co-clustering method arrives at two sets of eigen vectors, one that indicates a partitioning over the attributes, and a second that indicates a partitioning over the data objects.
- **Multi-view Relationship Learning:** Consider a specific task; now, across the different views associated with the task, we expect to see a consistency across the partitioning of data objects. This may be achieved by incentivizing for the similarity matrices (with each entry indicating pairwise object similarities) to be the same across the separate data partitioning models (i.e., eigenvectors) learnt from across the tasks.
- **Multi-task Relationship Learning:** Analogous to the previous consideration, this considers learning a shared subspace across related tasks, which is conformant to the data partitioning structure that entails from the eigenvectors learnt for the view-task combinations.

Each of the above considerations are modelled in an objective function, leading to an overall objective function for multi-view multi-task clustering. As in the previous cases, these separate considerations are optimized for an iterative framework, finally arriving at an multi-task MVC result.

2.11 Datasets for Multi-View Clustering

We now consider a few datasets for MVC that have been used in putting forward the empirical case for the various MVC methods that have appeared in the literature. Table 2.2 lists 27 datasets that have been used in order to evaluate MVC methods.

Table 2.2 Listing of a few datasets used for MVC

Dataset	$\|V\|$	Remarks	References
Reuters ML	6	Same text document in multiple natural languages	[30]
UCI Digits	2	Handwritten digits: views are Fourier coefficients Profile Correlations	[27]
IMDB	2	Movies represented across actors and keywords	[20]
CiteSeer	2	Scholarly articles with text and citation views	[20]
Cora	2	Scholarly articles with text and citation views	[20]
Cornell	2	Webpages with document and link views	[20]
Coral5k	2	Images with RGB histogram and SIFT views	[23]
Caltech101-7	6	Images with different types of image features	[46]
MSRC	6	Images with different types of image features	[46]
Handwritten	6	Images with different types of image features	[46]
WebKB	3	Web pages with text, inward anchor text, and title views	[48]
NUS-WIDE	7	Images with tags and six views of low-level features	[41]
VidTIMIT	2	Audio and video of a person speaking a sentence	[32]
Wikipedia	2	Text and inward/outward links	[32]
3-Sources	3	News Stories from BBC, Reuters, and The Guardian	[30]
ALOI	4	Four sets of image features	[24]
Pascal VOC	2	Images with color and bow features as views	[43]
SensIT	2	Transportation data with acoustic and seismic views	[7]
CQADupStack	2	Text data with question and answer views	[9]
Animal	6	Six sets of image features	[7]
SUN 397	7	Seven sets of image features	[7]
Water Treatment	4	Four sets of features of water treatment plants	[8]
Yeast Cell Cycle	5	Five sets of features from microarray data	[8]
Internet Ads	6	Image and text features associated with internet images	[8]
Yale	3	Three sets of image features	[50]
Notting-Hill	3	Three sets of image features from video face images	[50]
Oxford Flowers	4	Four sets of image features	[45]
SemEval2016-Task3	2	Text data with questions and comments as views	[10]

Many of the datasets have been used across multiple papers; however, for brevity, we just list one reference against each dataset. As may be seen from the table, the datasets comprise a wide variety of data types; text, links (e.g., weblinks, anchor texts, and citations), and image datasets. It is particularly worthy to note that many datasets have only two views, whereas a few datasets have as many as 6–7 views. However, even the six and seven views have been arrived at by partitioning features into sets of related features, rather than those views being intrinsic in the data representation itself. One may infer from the list that multi-view learning would benefit from the availability of more diverse datasets, such as those that come from varying domains, and those that organically have many views.

2.12 Conclusions

In this chapter, we have considered various families of approaches that have been explored for addressing the multi-view clustering problem. Starting with a formal definition of the problem, we considered the different formulations that have been employed in multi-view clustering, considering them in clusters of approaches based on similarities in their methodology. We then looked at some variations of the MVC setting that have been considered in the literature. We wish to re-emphasize here that our focus has not been to comprehensively cover techniques developed for multi-view clustering for there are too many of them; instead, we have chosen a few illustrative approaches from each school of methods in order to provide a diversified birds-eye view of methods for the task. Finally, we also listed a set of popular datasets that have been used for benchmarking and evaluating MVC algorithms. Our impetus has been to provide information in an accessible form, so that readers who may not be familiar with the mathematical details of specific machine learning building blocks would also be able to comprehend and utilize this chapter for scenarios such as choosing a particular MVC method for addressing a task at hand.

References

1. Balachandran, V., Deepak, P., Khemani, D.: Interpretable and reconfigurable clustering of document datasets by deriving word-based rules. Knowl. Inf. Syst. **32**(3), 475–503 (2012)
2. Bickel, S., Scheffer, T.: Multi-view clustering. In: ICDM, vol. 4, pp. 19–26 (2004)
3. Blei, D.M.: Probabilistic topic models. Commun. ACM **55**(4), 77–84 (2012)
4. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the Eleventh Annual Conference on Computational Learning Theory, pp. 92–100. ACM, New York (1998)
5. Borzsony, S., Kossmann, D., Stocker, K.: The skyline operator. In: 2001 Proceedings of the 17th International Conference on Data Engineering, pp. 421–430. IEEE, Piscataway (2001)

6. Cai, D., He, X., Han, J., Huang, T.S.: Graph regularized nonnegative matrix factorization for data representation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(8), 1548–1560 (2011)
7. Cai, X., Nie, F., Huang, H.: Multi-view k-means clustering on big data. In: IJCAI, pp. 2598–2604 (2013)
8. Chen, X., Xu, X., Huang, J.Z., Ye, Y.: Tw-k-means: automated two-level variable weighting clustering algorithm for multiview data. IEEE Trans. Knowl. Data Eng. **25**(4), 932–944 (2013)
9. Deepak, P.: Mixkmeans: clustering question-answer archives. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 1576–1585 (2016)
10. Deepak, P., Garg, D., Shevade, S.: Latent space embedding for retrieval in question-answer archives. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 855–865 (2017)
11. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B Methodol. **39**(1), 1–38 (1977)
12. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 269–274. ACM, New York (2001)
13. Ding, C., He, X., Simon, H.D.: Nonnegative Lagrangian relaxation of K-means and spectral clustering. In: European Conference on Machine Learning. pp. 530–538. Springer, Berlin (2005)
14. Ding, C.H., Li, T., Jordan, M.I.: Convex and semi-nonnegative matrix factorizations. IEEE Trans. Pattern Anal. Mach. Intell. **32**(1), 45–55 (2010)
15. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. **95**(25), 14863–14868 (1998)
16. Fred, A.L., Jain, A.K.: Combining multiple clusterings using evidence accumulation. IEEE Trans. Pattern Anal. Mach. Intell. **27**(6), 835–850 (2005)
17. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science **315**(5814), 972–976 (2007)
18. Hodge, V., Austin, J.: A survey of outlier detection methodologies. Artif. Intell. Rev. **22**(2), 85–126 (2004)
19. Hofmann, T.: Probabilistic latent semantic analysis. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, pp. 289–296. Morgan Kaufmann Publishers Inc., San Francisco (1999)
20. Hussain, S.F., Bashir, S.: Co-clustering of multi-view datasets. Knowl. Inf. Syst. **47**(3), 545–570 (2016)
21. Jain, A.K.: Data clustering: 50 years beyond k-means. Pattern Recogn. Lett. **31**(8), 651–666 (2010)
22. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs (1988)
23. Jiang, Y., Liu, J., Li, Z., Lu, H.: Collaborative PLSA for multi-view clustering. In: 2012 21st International Conference on Pattern Recognition (ICPR), pp. 2997–3000. IEEE, Piscataway (2012)
24. Jiang, B., Qiu, F., Wang, L.: Multi-view clustering via simultaneous weighting on views and features. Appl. Soft Comput. **47**, 304–315 (2016)
25. Jing, L., Ng, M.K., Huang, J.Z.: An entropy weighting K-means algorithm for subspace clustering of high-dimensional sparse data. IEEE Trans. Knowl. Data Eng. **19**(8), 1026–1041 (2007)
26. Kim, Y.M., Amini, M.R., Goutte, C., Gallinari, P.: Multi-view clustering of multilingual documents. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 821–822. ACM, New York (2010)
27. Kumar, A., Daumé, H.: A co-training approach for multi-view spectral clustering. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11), pp. 393–400 (2011)
28. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature **401**(6755), 788 (1999)

29. Liao, T.W.: Clustering of time series data—a survey. Pattern Recogn. **38**(11), 1857–1874 (2005)
30. Liu, J., Wang, C., Gao, J., Han, J.: Multi-view clustering via joint nonnegative matrix factorization. In: Proceedings of the 2013 SIAM International Conference on Data Mining, pp. 252–260. SIAM, Philadelphia (2013)
31. Liu, H., Liu, T., Wu, J., Tao, D., Fu, Y.: Spectral ensemble clustering. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 715–724. ACM, New York (2015)
32. Livescu, K., Sridharan, K., Kakade, S., Chaudhuri, K.: Multi-view clustering via canonical correlation analysis. In: NIPS Workshop: Learning from Multiple Sources (2008)
33. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Oakland, vol. 1, pp. 281–297 (1967)
34. Meng, X., Liu, X., Tong, Y., Glänzel, W., Tan, S.: Multi-view clustering with exemplars for scientific mapping. Scientometrics **105**(3), 1527–1552 (2015)
35. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: analysis and an algorithm. In: Advances in Neural Information Processing Systems, pp. 849–856 (2002)
36. Nie, F., Huang, H., Cai, X., Ding, C.H.: Efficient and robust feature selection via joint $\ell 2$, 1-norms minimization. In: Advances in Neural Information Processing Systems, pp. 1813–1821 (2010)
37. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)
38. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
39. Tao, Z., Liu, H., Li, S., Ding, Z., Fu, Y.: From ensemble clustering to multi-view clustering. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI), pp. 2843–2849 (2017)
40. Thompson, B.: Canonical correlation analysis. In: Encyclopedia of Statistics in Behavioral Science. Wiley, West Sussex (2005)
41. Wang, H., Nie, F., Huang, H.: Multi-view clustering and feature learning via structured sparsity. In: International Conference on Machine Learning, pp. 352–360 (2013)
42. Wang, X., Qian, B., Ye, J., Davidson, I.: Multi-objective multi-view spectral clustering via pareto optimization. In: Proceedings of the 2013 SIAM International Conference on Data Mining, pp. 234–242. SIAM, Philadelphia (2013)
43. Wang, D., Yin, Q., He, R., Wang, L., Tan, T.: Multi-view clustering via structured low-rank representation. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 1911–1914. ACM, New York (2015)
44. Wang, C.D., Lai, J.H., Philip, S.Y.: Multi-view clustering based on belief propagation. IEEE Trans. Knowl. Data Eng. **28**(4), 1007–1021 (2016)
45. Wang, Y., Chen, L., Li, X.L.: Multiple medoids based multi-view relational fuzzy clustering with minimax optimization. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, pp. 2971–2977 (2017)
46. Xu, J., Han, J., Nie, F.: Discriminatively embedded K-means for multi-view clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5356–5364 (2016)
47. Xu, Y.M., Wang, C.D., Lai, J.H.: Weighted multi-view clustering with feature selection. Pattern Recogn. **53**, 25–35 (2016)
48. Zhang, X., Zhang, X., Liu, H.: Multi-task multi-view clustering for non-negative data. In: IJCAI, pp. 4055–4061 (2015)
49. Zhang, X., Zong, L., Liu, X., Yu, H.: Constrained NMF-based multi-view clustering on unmapped data. In: AAAI, pp. 3174–3180 (2015)
50. Zhao, H., Ding, Z., Fu, Y.: Multi-view clustering via deep matrix factorization. In: AAAI, pp. 2921–2927 (2017)
51. Zong, L., Zhang, X., Zhao, L., Yu, H., Zhao, Q.: Multi-view clustering via multi-manifold regularized non-negative matrix factorization. Neural Netw. **88**, 74–89 (2017)

Chapter 3
Semi-supervised and Unsupervised Approaches to Record Pairs Classification in Multi-Source Data Linkage

Anna Jurek-Loughrey and Deepak P

Abstract Data integration has become one of the main challenges in the era of Big Data analytics. Often to enable decision-making, data from different sources have to be integrated and linked together. For example, multi-source data integration is vital to police, counter terrorism and national security to allow efficient and accurate verification of people. One of the key challenges in the data integration process is matching records that represent the same real-world entity (e.g. person). This process is referred to as record linkage. In many cases, data sets do not share a unique identifier (e.g. National Insurance Number), hence records need to be matched by comparing their corresponding attributes. Most of the existing record linkage methods require assistance from a domain expert for handcrafting domain-specific linking rules. More automatic approaches, based on using machine learning, were also proposed. However, those approaches relay on having a substantial set of manually labelled records, which makes them inapplicable in real-world scenarios. Given the importance of the problem, record linkage has witnessed a strong interest in the past decade. As a result, significant progress has been made in this area. In particular, the problem of reducing the manual effort and the amount of labelled data required for constructing record linkage models has been addressed in many studies. In this chapter, we review the most recently proposed approaches to semi-supervised and unsupervised record linkage.

A. Jurek-Loughrey (✉) · Deepak P
Queen's University Belfast, Belfast, UK
e-mail: a.jurek@qub.ac.uk; deepaksp@acm.org

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_3

3.1 Introduction

Society worldwide is generating more and more data giving rise to the "data deluge" problem. Making sense of such data is necessary for making strategically important decisions by government bodies, security, healthcare and financial entities, to name a few. In the world of big data, data integration technology is crucial for maximising the capability of data-driven decision-making. Integrating data from multiple sources drastically expands the power of information and allows us to address questions that are impossible to answer using a single data source. For example, analysis of the health data coming from sources, such as electronic health records, drug and toxicology databases, genomics and social media environments, is a key driver toward advancing precision medicine. As a part of the data integration process, records (from two or more data sources) that refer to the same real-world entity (e.g. person) need to be linked. This process is referred to as record linkage (RL), data/record matching and entity resolution. In many realistic scenarios, different data sources do not share a unique identifier (e.g. National Insurance Number), thus the process of linking records needs to be performed by matching their corresponding attributes. This becomes challenging due to the heterogeneous character of the data. Records in different data sources are often represented by different number and different type of attributes. The same data can also be represented in different ways in different data sources due to factors such as different conventions (e.g., International Conference on Management of Data versus SIGMOD Conference). Furthermore, data quality is often affected by typographical errors, and missing and out-of-date values. As a consequence, it is not always the case that records referring to the same entity have the same values on the corresponding attributes. Furthermore, the data may be unstructured (e.g. text reports, and social media content) or have different formats (e.g. text, image and video). Therefore, more sophisticated techniques are required for effective RL. Commonly applied RL methods require assistance from a domain expert to carefully handcraft bespoke domain-specific linking rules that aid in determining the linkage likelihood of a candidate record pair. This requires deep topical expertise in the domain and continuous maintenance to cope with any changes in the character of the data, which is a costly proposition in many realistic scenarios. Probabilistic RL methods [10] compute weights for different attributes based on their estimated ability to correctly identify matching and non-matching records. The weights are then applied to calculate the probability that two given records represent the same real-world entity. Probabilistic methods relay on estimated probabilities and thresholds, which requires some domain knowledge. More automatic RL techniques have been proposed; however, they require a large amount of labelled data to train machine learning models. They also rely on the selection of similarity measures and attributes (matching schemes) applied for comparing pairs of records. This makes RL very expensive and labour intensive, hence it is not affordable for many organisations. Given its pivotal importance and challenges, there has been strong interest in semi-supervised and unsupervised approaches to RL in

the last decade within the computer science domain. RL methods, which address issues such as data labelling, user effort or heterogeneity of the data, have been explored.

Overview of the Paper This chapter gives an overview of the most recent advances in the area of semi-supervised and unsupervised RL. It focuses on the task of comparing and classifying pairs of records from different sources. It considers different families of techniques that have been explored in the last few years to address the problem of generating RL models with limited training data. The chapter will be useful for readers who are trying to decide on the most appropriate RL model for their data, which requires minimum manual effort from the user.

3.1.1 Overview of the Record Linkage Task

Given two sets of records S and T, RL is defined as a task of identifying pairs of records $(s, t) \in S \times T$ that refer to the same entity (e.g. a person) [31]. Given two data sources, each pair of records (s, t) can be classified into one of the two classes: match and non-match. Table 3.1 shows a simple example of an RL task. The table contains records from two bibliographic data sources (DBLP, and ACM digital library). The aim is to identify those pairs of records that refer to the same publications, which in this case are (ACM1, DB1) and (ACM2, DB2). Any other pairs of records should be identified as non-match. RL has been widely applied in data management, data warehousing, business intelligence, historical data collection and medical research [7]. Figure 3.1 illustrates an overview of an RL process, which is composed of four main steps.

Table 3.1 An example of RL

ID	Title	Authors	Venue
ACM1	A compact B-tree	Peter Bumbulis, Ivan T. Bowman	International conference on management of data
ACM2	A theory of redo recovery	David Lomet, Mark Tuttle	International conference on management of data
DB1	A compact B-tree	Ivan T. Bowman, Peter Bumbulis	SIGMOD conference
DB2	A theory of redo recovery	Mark R. Tuttle, David B. Lomet	SIGMOD conference
DB3	Enhanced abstract data types in object-relational databases	Praveen Seshadri	VLDB J.
DB4	Parametric query optimization	Raymond T. Ng, Timos K. Sellis, Yannis E. Ioannidis, Kyuseok Shim	VLDB J.

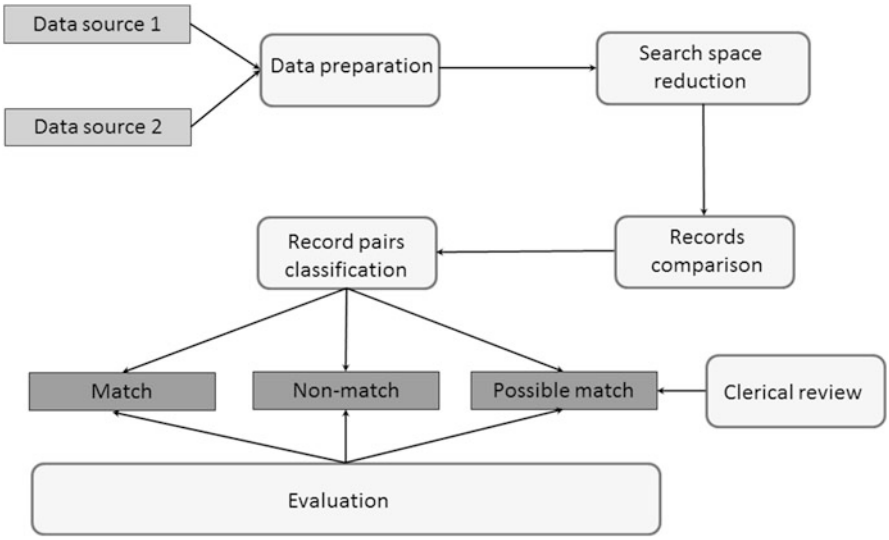


Fig. 3.1 Record linkage process

Data preparation aims to make the records coming from different data sources comparable. The process includes steps such as parsing, data transformation, data standardisation and attribute matching, which have been recognised as crucial steps in the RL process [14].

Search space reduction focuses on improving the speed of the RL process through reduction of the number of record comparisons. If we need to link two data sets, S and T, then potentially we should compare each record from S with each record from T. Hence, the total number of comparisons would be $|S| \times |T|$. To reduce the potentially large number of record comparisons, different forms of indexing and filtering, collectively referred to as blocking [2, 8, 39], are deployed in RL systems. The idea of blocking is to use a blocking function to divide records into a set of blocks. The candidate pairs of records for linkage are then selected from records within the same block only.

Record comparison is a process of determining similarity between any candidate pair of records provided as an output of the blocking process. Comparing records relies strongly on the use of a similarity measure, in which selection of such measures is a key issue [12]. Given two records s and t represented by N attributes $f_1, \ldots, f_N$, a similarity measure m quantifies similarity between the two records on one of the attributes. It returns a numeric value ranging between 0 and 1, referred to as a similarity value. A similarity measure is formulated as $m : dom(f_i) \times dom(f_i) \rightarrow [0, 1]$, where $dom(f_i)$ denotes the domain of attribute f_i. Value 1 indicates that the pair of records have the exact same values on an attribute according to the similarity measure. Value 0 indicates that there is no similarity between the attributes values for the two records. There are many available similarity measures,

e.g. Jaro [21], Jaro-Winkler [48], Jaccard [19], Q-Gram [36] and Levenshtein edit distance [27]. Depending on the types of data in RL, similarity measures have different levels of accuracy [9] and as emphasised in [3], there does not exist a single similarity measure for all data sets.

Record pair classification is the process of classifying the compared pairs of records as either match or non-match. The existing approaches can be broadly divided into two categories. The first category of approaches rely on applying generic linkage rules (LR) and different similarity measures to identify those pairs of records that are similar enough to be considered as matching [15, 42]. Notationally:

$$[S, T, M] \Longrightarrow LR : S \times T \rightarrow 0, 1 \tag{3.1}$$

where M refers to a set of similarity measures, and 0, 1 indicate match and non-match, respectively. With the second category of approaches, each pair of records is represented as a comparison vector with N elements. Given two records r_1 and r_2 represented by N attributes $f_1, \ldots, f_N$ and a similarity measure m, a comparison vector for s and t is formulated as:

$$m(s, t) =< m(r_1.f_1, r_2.f_1), \ldots, m(r_1.f_N, r_2.f_N) > \tag{3.2}$$

The comparison vector represents a set of N numeric similarities, each calculated with a similarity measure on the corresponding pair of attribute values of the two records. The task of RL is then considered as a comparison vector classification problem [11], i.e., whether a comparison vector is classified as match or non-match, for a pair of matching or non-matching records, respectively. Notationally:

$$[S, T, m] \Longrightarrow LR : \overrightarrow{V} \rightarrow 0, 1 \tag{3.3}$$

where $\overrightarrow{V} = \{m(s_i, t_j) : s_i \in S, t_j \in T\}$ is the set of comparison vectors generated for each record pair from $S \times T$.

With learning-based RL methods (e.g. applying statistical or machine learning techniques), a linkage model is learnt from training data and then applied to link new pairs of records. The training data set has the following format:

$$D = \{(s_1, t_1, l_1), (s_2, t_2, l_2), \ldots, (s_n, t_n, l_n)\} \tag{3.4}$$

where each triple consists of a record pair (s_i, t_i) and a label l_i with value 1 or 0 if the two records match or not, respectively.

In this chapter, we focus on the problem of classifying pairs of records based on their similarity, which is one of the primary fields of research in RL. A large number of effective supervised approaches to RL have been proposed in the past [7, 12]. Supervised approaches require labelled training data with known matching status of the record pairs. In particular, a substantial collection of comparison vectors labelled as match or non-match is required in order to train a supervised model. Obtaining high-quality training data in a real-world scenario is challenging or even impossible

in some cases. In the next section, we will review some of the recently proposed approaches to RL that allow us to relax or completely remove the requirement for labelled data.

3.1.2 Overview of Existing Semi-supervised and Unsupervised Record Linkage Techniques

In order to address the problem of lack of labelled data, various semi-supervised and unsupervised learning techniques have been proposed for RL over the past few years [7]. In semi-supervised learning, only a small set of labelled instances and a large set of unlabelled instances are used in the training process. With unsupervised approaches, no labelled data, in the form of record pairs with known matching status, is required for generating the decision model. Below, we give a brief overview of the most recent semi-supervised and unsupervised approaches to RL.

- *Crowdsourced-based methods.* Given the availability of easily accessible crowdsourcing platforms such as Amazon Mechanical Turk and Crowdflower, there has been a lot of interest in the human-in-the-loop approaches to RL. Those approaches leverage the crowd's ability to solve the RL problem. In order to minimise the number of questions sent to the crowd, hybrid human–machine systems have been explored. Those models utilise machines to select a subset of record pair candidates, which are then sent to the crowd for labelling. The feedback obtained from the crowd is then utilised to label the remaining pairs.
- *Active learning.* Similar to crowdsourcing RL, this is a human-in-the-loop-based approach. The aim of active learning is to reduce the number of record pairs that need to be labelled via actively selecting the most informative examples. Active learning algorithms select a small subset of record pairs which are then sent for manual labelling to a user. The obtained labels are further utilised in the learning process. The process is repeated until all examples are labelled.
- *Iterative self-learning.* Given a training data set $L = \{x_i, y_i\}_{i=1}^{l}$ (referred to as seeds) and an unlabelled data set $U = \{x_j\}_{j=l+1}^{l+u}$ (usually, $L \ll U$), the self-learning process aims to train a classifier h on L initially and then use the classifier to label all unlabelled data in U. Following this, the most confidently classified unlabelled examples are selected as new training examples and moved into L. The process is repeated until a stopping condition is met.
- *Graph-based models.* Those approaches use graphs to model record's structure. The nodes in the graphs represent records and their properties. The edges represent the relation between them. The goal is to estimate the likelihood of two nodes being connected with an edge in a graph.
- *Bayesian models.* Those approaches represent the relations between records as a bipartite graph, in which records are linked to latent true variables. They assume conditional independence between records, given the latent individuals,

and independence between the attributes values within individuals. The goal is to estimate the probability that two records are linked to the same latent true variable.
- *Self-learning with automatically generated seeds.* Those methods are based on the same idea as the aforementioned self-learning-based approaches, with the difference that they automatically generate their own set of labelled record pairs, which are then used as seeds to start the learning process.
- *Optimisation of an objective function.* With those approaches, an objective function used to evaluate the performance of an RL model using only unlabelled data is formulated. Following this, a searching algorithm is applied to find the RL model which maximises/minimises the objective function.
- *Knowledge graph embedding.* In the most recent work, knowledge graph embeddings have been proposed for solving the RL problem. The basic idea is to embed entities and their relations into a low-dimensional vector space, which can then be used for matching entity pairs.

3.1.3 Existing Publicly Available Data Sets for Evaluation of Record Linkage Models

Below, we describe the most common data sets that are used by researchers working in the field of RL for evaluation and comparison of different RL models. Some of the data sets contain records from only one data source, which is appropriate for the deduplication task. Most of the data sets are publicly available.

- *Restaurant*[1] A collection of 864 records from the Fodor's and Zagat's restaurant guides. The task is to find records referring to the same restaurant. Each record is represented by five attributes: name, phone number, address, city and cuisine. The data set contains 112 duplicates.
- *Cora* (see footnote 1) The data set contains a collection of 1295 bibliographic records of research papers on machine learning. Citations of the same paper may have a different format; therefore, the task here is to recognise citations of the same paper. Each of the records contains the publication's author name, paper's title, name of the venue where the paper was published and the date. The data set contains 17,184 matching pairs.
- *DBLP-ACM*[2] Data set containing bibliographic records from 2 data sets of size 2616 and 2294. In total, there are 2224 unique entities. Each record is represented by four attributes: title of the paper, authors' names, name of the venue and the year of publication.

[1] http://www.cs.utexas.edu/users/ml/riddle/data.html.

[2] https://dbs.uni-leipzig.de/en/research/projects/object_matching/fever/benchmark_datasets_for_entity_resolution.

- *DBLP-Scholar* (see footnote 2) This data set contains bibliographic records of 2 data sets of size 2616 and 65,263 with 5347 matching pairs. The format of the records is the same as the DBLP-ACM data set.
- *Amazon-Google Products* (see footnote 2) The data set contains product records from 2 data sets of size 1363 and 3226, respectively. There are a total of 1300 matching record pairs. The records are represented by four attributes including product name, product description, name of the manufacturer and the price of the product.
- *Abt-Buy* (see footnote 2) A collection of product records from two data sources of size 1081 and 1092. The format of the records is the same as with Amazon-Google Products. There are 1097 unique entities in total.
- *Census*[3] This is a collection of synthetic census records generated by the US Census Bureau. There are two data sets with 449 and 392 records, respectively. The number of matching record pairs is 327. Each record contains first and last name, middle initials, a street number and a street name.
- *CDDB*[4] A collection of 9763 records of CD albums containing information such as artist name, title, genre and the year when the album was published. There are 6087 matching record pairs.

3.1.4 Evaluation of Record Linkage Quality

The quality of an RL model can be evaluated assuming that some data sets with truly labelled record pairs as match or non-match are available. Most of the RL data sets are very unbalanced with a much larger fraction of non-matching record pairs. Given the lopsided distribution, the number of record pairs which were correctly classified by the model as non-matches dominates its final evaluation. For this reason, the $accuracy$ measure is not suitable for the evaluation of RL models. Consider the case of $Restaurant$ data set, where there only could be 112 record pairs correctly classified as match and 372,704 pairs that can be correctly labelled as non-match. A simple classification of all record pairs as non-matches will result in a very high accuracy, which clearly will not be meaningful. Consequently, in majority of the papers the authors use F-measure as the evaluation measure for RL quality. F-measure is the harmonic mean between precision and recall and only has high value if both precision and recall are high.

[3] https://www.census.gov/data/datasets.html.

[4] http://www.freedb.org/en/download__server_software.4.html.

3.2 Semi-supervised Record Linkage

In order to address the problem of lack of labelled data, various semi-supervised techniques have been proposed for RL, over the past few years. Those approaches try to relax the assumption on availability of a substantial training set. They usually only require a small set of labelled examples, which is then utilised to label the remaining record pairs. In the following sections, we review the most recently proposed semi-supervised RL methods.

3.2.1 Crowdsourced-Based Methods

The limitations of learning-based approaches and the existence of easily accessible crowdsourcing platforms lead to the development of many human-in-the-loop-based approaches. Crowdsourcing platforms allow us to task people with small jobs, such as text or image labelling, via a programmatic interface. The idea of crowdsourced-based RL is to use a crowd of people for identifying whether a record pair represent the same real-world object. A brute-force method based on crowdsourcing enumerates every record pair and sends them to the crowd. Obviously, for a large data set this would involve huge monetary costs and waiting time. Different studies tried to address this problem in order to avoid the exhaustive comparison of all the records. In [30], the authors proposed a batching strategy where multiple pairs of records are placed in a single request. It was demonstrated that batching can significantly improve the efficiency with only small loss in accuracy. However, such an approach still suffers from a scalability problem even for a modest data set size hence it is not sufficient to enable RL to be done at scale. To reduce the number of questions being sent to the crowd, a number of hybrid human–machine approaches were proposed. The research in this area is focused on two main challenges. First one is the development of intelligence machine-based techniques for ruling out obvious non-matching pairs and selecting an optimal set of questions to minimise the monetary cost. Second challenge is how to utilise labels obtained from the crowd for automatic labelling of the remaining record pairs.

In [43], a technique, which applies transitive relations between records to reduce the number of crowdsourced record pairs, is proposed. All records are modelled as nodes in a graph with edges representing labels for pairs of records. The authors formulate two transitive relations that allow us to deduct some of the missing edges in the graph. According to the positive transitive relations, if there exists a path between two records which only consists of matching pairs, then the pair of records can also be deducted as a match. According to the negative transitive relation, if there exists a path between two records which contains a single non-matching pair, then the pair of records can be deducted as a non-matching. In other cases, the label for the two records cannot be deducted. In their work, the authors observed that labelling a matching pair at first will lead to fewer or equal number of crowdsourced pairs. Following this observation, they apply a sorting algorithm which aims to

identify labelling order that minimises the number of crowdsourced pairs. The two labelling processes, based on crowdsourcing and based on transitive relation deduction, are performed in parallel. Any labels obtained from a crowdsourcing platform are instantly utilised to deduct unlabelled pairs. It has been demonstrated that such an approach can reduce the cost with little loss in the results quality. One obvious drawback of this approach is the assumption that the crowd always returns a correct result. When errors occur in the results obtained from the crowd, the transitive relation fails and tends to amplify it. For instance, suppose that $r_1 = r_2$ and $r_2 \neq r_3$, but the crowd returns $r_1 = r_2$ and $r_2 = r_3$. Consequently, we can incorrectly deduct that $r_1 = r_3$.

A hybrid human–machine framework for entity resolution, which also utilises the transitive relations, is proposed in [28]. The first phase of the proposed model is pruning of a large number of dissimilar pairs. For this purpose, a set of similarity-based rules is generated using some given positive and negative examples. The rules are then applied with a distributed in memory system in order to select a set of candidate record pairs. In the second phase, a selection inference refine framework is applied to verify the candidate pairs. The framework first selects tasks (record pairs) for the crowd. In order to minimise the monetary costs, the tasks are first ordered in a way that the pairs with higher probability of being a match are sent to the crowd first. The probability is estimated using the similarity between the records. The answers from the crowd are then used to answer unasked tasks using the transitivity assumption. The process is repeated until all candidate record pairs are labelled. To address the human error issue, they developed a truth inference system for crowdsourcing [50], which implements most of the existing inference algorithms. Truth Inference aims to infer the truth for each task while considering the quality level of each worker. The framework in [28] can automatically infer the quality of workers and hence identify a high-quality answer for each task what leads to better linking accuracy.

An approach based on correlation-clustering, which tries to address the sensitivity to the human errors problem, is proposed in [47]. The overall idea of this work is to divide all records into disjoint clusters, such that any two records belonging to the same cluster represent the same entity. First, a similarity score function is constructed, which together with a threshold is used to prune dissimilar record pairs. Following this, a selected set of record pairs is sent to the crowd for labelling. Each record pair is evaluated by multiple crowd workers. Crowd-based similarity score is defined as a percentage of crowd workers who believed that two records refer to the same entity. Based on the crowd-based similarity score, the records are split into a number of disjoint clusters. All the record pairs eliminated in the pruning process automatically obtain a score of zero. Finally, the cluster refinement phase is performed, where a number of selected pairs are issued to the crowd to check if their answers are consistent with the clusters. If many of the answers contradict with the clustering, the clusters are adjusted. Using multiple workers for each record pair and the refinement phase help to address the problem of human error. It however requires a larger number of questions to be asked, which increases the monetary costs.

In order to further reduce the monetary costs, a partial-order-based crowdsourced entity framework is proposed in [5]. The authors define a partial order, where one pair of records $(r_i{}^1, r_j{}^1)$ precedes another pair $(r_i{}^2, r_j{}^2)$ if $(r_i{}^1, r_j{}^1)$ has no smaller similarity than $(r_i{}^2, r_j{}^2)$ on every attribute. All pairs of records are modelled as a graph based on the partial order, where each vertex represents a pair and directed edge from $(r_i{}^1, r_j{}^1)$ to $(r_i{}^2, r_j{}^2)$ indicates that the former precedes the latter. Selected vertex in the graph is sent to the crowd for labelling. If a vertex is labelled as a match, all the pairs preceding this pair are automatically labelled as a match. If a vertex is labelled as a non-match, all succeeding pairs are automatically label as a non-match. To further reduce the number of questions, similar vertices are grouped together. A path-based pair selection algorithm is proposed to minimise the number of questions to be asked. In order to address the possible errors coming from the crowd, each vertex is labelled by multiple workers. For the vertex with low confidence, an additional weighted similarity is applied before the vertex is labelled as match or non-match. The proposed partial-order allows to further reduce the candidate pairs delivered for crowd task, and it has been demonstrated that it can obtain superior accuracy in comparison to other approaches. However, it might not be applicable with noisy data sets, where the preceding condition does not hold for many record pairs.

The crowdsourced-based RL systems do not require any data for training, and according to the presented results, they can achieve superior accuracy to some other semi-supervised and unsupervised approaches. However, those methods can be costly for large data sets and also require a long waiting time on the feedback from the crowd workers. Each of the aforementioned methods assumes that the data is structured and represented by an aligned set of attributes. Some of the methods perform record comparisons, either for pruning or automatic labelling of record pairs. However, the problem of selecting matching schema used for comparing records is not addressed in any of the paper.

3.2.2 Learning-Based Approaches

With the learning-based approaches, RL is considered as a comparison vector classification problem, where each vector represents similarity values between a record pair calculated with a similarity measure. They rely on training data sets to train a classification model. In order to address the limitations of supervised RL models, various semi-supervised learning techniques have been proposed for RL over the past few years. Those methods require only a small subset out of all the given record pairs to be labelled as match or non-match. A popular and promising approach to semi-supervised RL is the one based on active learning (AL) [1]. The basic idea of AL is to reduce the labelling effort by interactively querying the user for labels of the most informative examples. Instead of labelling a large number

of examples, as in the case of supervised learning, AL selects a small subset to be labelled by the user. The selection is made based on the examples labelled in the previous iteration.

In [46], the authors propose an interactive training model based on AL. With this approach, all the record pairs are clustered by their comparison vectors. Following this, a number of comparison vectors from each cluster are selected and their corresponding pairs of records are then manually classified by a user as match or non-match and added to the final training set. Based on the feedback received from the user, the purity of each cluster is calculated. The purity is formulated as the proportion of classified comparison vectors that have the majority label. For each cluster, if the purity is greater than a predefined threshold, the remaining comparison vectors from the cluster are automatically labelled as match or non-match and added to the training data set. Otherwise, the cluster is further split into sub-clusters. The process is repeated recursively until the total budget for manual labelling has been fully used. The crucial factor of this approach is the selection of the most informative comparison vectors within the clusters. The authors explore a number of different selection methods, including random selection, selection of vectors that are farthest away within the cluster and combination of the farthest apart vectors with the vector closest to the centre of the cluster. It is demonstrated that the approach obtains comparable results to supervised models, yet it requires significantly lower runtime and labelling effort. The performance of the model relies on the selection of a number of parameters including budget size, number of vectors selected from each cluster in the AL process and the purity threshold. Those may be hard to determine without any labelled data provided before the learning process begins. The method also relies on the selection of similarity measures and attributes used for generating the comparison vectors.

The application of AL and genetic programming for learning linking rules was explored in [16] and [25]. A linkage rule is defined as a similarity function (which assigns a similarity value to each record pair) and a threshold. A pair of records is considered as matching according to a linkage rule, if the similarity value exceeds the threshold. With [16] and [25], linkage rules are represented as trees built with different types of operators. In [16], they use similarity operators, which evaluate the similarity between values of two inputs using a similarity measure, aggregation operators for aggregating the similarity scores and value operators which retrieve values of a specific attribute or a transformation of those values (e.g. normalisation, and tokenisation). The idea of the algorithm is to construct a population of candidate linkage rules iteratively while building a set of reference links between records. The algorithm begins with a random population of linkage rules and an empty set of links. As part of this process, compatible pairs of attributes are identified based on their values overlap. There are three steps involved in each iteration: (1) Selection of the link candidates to be manually labelled by a user from a pool of unlabelled entity pairs. The selected links are those that reduce the version space of the linkage rules from the entire population. (2) Manual labelling of the selected links as correct or incorrect, which are then added to the set of reference links. (3) Evolving of the population, which is based on the current reference links, by creating new linkage

rules from the existing population using the genetic operators. The evolving process is continued until a linkage rule that covers the current set of labelled links is found.

A similar idea of using genetic algorithm and AL for finding the most appropriate linkage rule is presented in [32]. The link specifications are formulated as trees, which combine different atomic link specifications. An atomic link specification uses a single similarity measure to determine similarity between two attributes' values, which is then compared with a threshold. A pair of atomic link specifications can be combined via operators such as AND, OR and XOR. The algorithm begins with generating a random population of link specifications. All individuals in the population are then applied to link instances from two data sources. The pairs of records on which the individuals disagree the most are then selected for manual labelling. The fitness of each individual is calculated using F-measure on the available labelled links. The link specifications are evolved using the genetic operators: reproduction, mutation and crossover.

Application of genetic algorithms and AL reduces the requirement for labelled data. It also allows us to address the problem of selecting an optimal matching scheme, since multiple similarity measures and attributes can be used by different linkage rules. However, genetic algorithms have to check many different configurations to reach the convergence, which may not scale with large data sets.

Two different approaches to RL based on semi-supervised learning are proposed in [25] and [37]. Both of the methods require a small set of record pairs, labelled as match or non-match, to start the training process. The system in [25] is based on the idea of self-learning. The set of labelled examples provided as an input is used to initially train a classification model. The initial classification model is then applied to classify the unseen data. A small percentage of the most confidently classified record pairs is selected to iteratively train the classification model. The process is repeated for a number of iterations, or until all the unseen data are labelled. To maximise the performance of the model on unseen data, an ensemble technique referred to as boosting is employed together with weighted Majority Voting. The system relaxes the assumption on substantial training data set required for constructing the classification model. It addresses the learning performance problem by applying ensemble techniques to improve the accuracy. However, it requires a number of parameters to be specified, including the maximum number of iterations, and the percentages of the most confident matching and non-matching examples to be selected in each iteration. It also relies on the selection of similarity measures.

In [37], a links learning algorithm that requires only positive examples is proposed. The authors refer to the fact that the public interlinked RDF data on the Web provide only positive links and no negative examples are available. Therefore, for a supervised machine learning method to be applied for link discovery, the user has to provide the algorithm with negative examples. With the proposed approach, a set of initial atomic link specifications that achieve the highest possible F-measure for all known record pairs is selected. The link specifications are constructed using different combinations of similarity measures and attributes. After deriving atomic link specifications, an iterative search through a solution space is performed with

F-measure as the score function, in order to compute more complex specifications. This work somehow relaxes the assumption on labelled data. However, in many realistic scenarios the positive links between records are not available. In fact, in majority of cases, it is much easier to obtain negative links among records. Therefore, the approach has limited application to realistic scenarios.

3.3 Unsupervised Record Linkage

Semi-supervised learning significantly reduces the number of manually labelled examples required for generating an RL model. However, it still requires a certain amount of human input in the training process. In this section, we provide an overview of recently proposed approaches that try to completely remove the requirement for labelled training data.

3.3.1 Graph-Based Models

A group of unsupervised models for collective RL, which use graphs to model record's structure, were proposed. In [51], RDF data are modelled as multi-type graphs, and the collective RL is formulated as a multi-type graph summarisation problem. With the multi-type graph, the vertices represent different types of properties (e.g. name, and title), and edges represent co-occurrence between two properties or similarity between two properties of the same type. The similarities between the vertices are calculated using a similarity measure. The RDF graph is then converted into a summary graph, where the nodes representing different mentions of the same entity belong to the same super node. The graph summarisation is considered as an optimisation problem, which accommodates two types of similarities between vertices. Inherent similarity described by the content of the nodes (string similarities between their labels) and structural similarity reflecting connectivity patterns in the multi-graph. The algorithm starts with a random summary graph. The number of super nodes is fixed via backward search procedure, which repeatedly removes one or many super nodes from the summary graph with the lowest information. Once the number of super nodes is fixed, the assignment (of vertices to super nodes) is optimised. The procedures are repeated until a locally optimal summary graph is found.

In [49], a framework that takes into account content-based similarity and topological structure, to estimate probability that two records refer to the same entity, is proposed. The framework is composed of two main components. First component is responsible for generating a weighted bipartite graph between terms and pairs of records. The graph represents the importance of terms and similarities of record pairs. Two records are considered similar if they contain similar terms. A recursive formula is applied for estimating the term's weights and record's similarities, where

a term is assigned with a high weight if all the record pairs sharing the term have high similarity. The similarity between two records is calculated as the sum of normalised common terms' weights. The second component is responsible for constructing a weighted record graph to estimate the likelihood of two records referring to the same entity based on a similarity function. The likelihoods are estimated with a customised version of the standard random walk. The estimated likelihoods are then applied to update the terms' weights in the bipartite graph until the join optimum can be found. The experimental evaluation showed that without any labelled data the approach can be comparable to the crowd-assisted methods.

3.3.2 Bayesian Models

Recently, Bayesian methods and latent variable modelling have been explored in unsupervised RL. In [38, 41], a Bayesian approach based on a parametric model for categorical data is proposed. With this approach, matching and non-matching records are modelled as a bipartite graph, with edges linking an observed record to the latent entity to which it corresponds. All latent entities have true values for their attributes and the attribute values of the associated records can be distorted from the true values with some probability. The goal is to estimate the posterior distribution of the linkage, which can be considered as a process of clustering similar records to hypothesised latent entity. The model is estimated with a hybrid Markov chain Monte Carlo algorithm [20]. It assumes conditional independence of records and different attributes for the same record. Such a linkage structure allows for unified representation of multi-source RL and deduplication under a single framework. The model can only be applied to categorical data, which does not take under consideration the distance between two different string representations that do not exactly agree (e.g. different version of the same name). The Bayesian model also requires the specification of the priors for the latent entities' values, which may be difficult to specify in particular for string values. Further advancements to this model are made in [35]. The authors extend the model, to both categorical and string values, using a coreference matrix. Similarly in [40], the model was extended to incorporate non-categorical data using the empirical distribution of the data for each attribute as the prior for the corresponding attribute values of the latent entities. An advantage of Bayesian models is that they handle the uncertainty quantification for the resulting estimates.

In [18], a method based on cluster matching is proposed to tackle the unsupervised RL of multiple-type network data. The approach assumes that different networks have common latent clusters, which have a specific interaction pattern to other clusters (e.g. synonym groups in multiple lexical networks in multiple languages or matching user communities in different friendship networks). The goal is then to find matching clusters from different networks (unsupervised cluster matching). A non-parametric Bayesian model is applied to cluster objects (nodes) in different networks into shared set of clusters, assuming that interaction patterns

between clusters are shared among different networks. Objects assigned to the same clusters within a network are considered as matching. The cluster number is automatically inferred using Dirichlet process. With the proposed method, every object is assumed to be assigned into a cluster; however, in real-world scenarios some objects may not be relevant. In [17], the model was expanded to address this problem. The model estimates the cluster assignments and the relevance of objects simultaneously. Consequently, only the objects that are relevant from the clusters are considered, which improves the matching performance.

3.3.3 Learning-Based Approaches

Unsupervised learning methods, which allow us to train a classification model without any ground truth data, have also been explored in the field of RL. An approach that received a lot of interest is the one based on automatic self-learning. Automatic self-learning methods eliminate the requirement for labelled data to be provided as an input by generating its own initial training set. In [6], an approach to automatic seed selection, referred to as nearest based, is applied. With the nearest-based approach, all comparison vectors are sorted by their distances (e.g. Manhattan or Euclidean distance) from the origin $[0, 0, \dots]$ and from the vector with all similarities equal to 1 ($[1, 1, \dots]$). Following this, the respective nearest vectors are selected as match and non-match seeds. The number of match and non-match seeds to be selected are taken as the input parameters to the method. The authors evaluate three sizes of the non-match seeds, namely 1%, 5% and 10% of the entire data set. Following this, an appropriate ratio of similarity vectors is selected as match seeds. After selecting seeds, a classification model is trained incrementally following the self-learning procedure. In the same work, it was observed that this approach did not provide good quality match seeds when the data set contained only a small number of matching records or there were some fields with a large proportion of abbreviations.

In [26], an unsupervised system based on automatic self-learning for matching entities in schema-free RDF files is proposed. The overall idea of the approach is to generate its own heuristic training set, which can then be used for learning a link specification function. In the training set generator process, each RDF file is converted into bag-of-words document. Following this, the term frequencies and inverse document frequencies together with a threshold are applied to eliminate the obvious non-matching pairs. All selected pairs are further sorted in descending order according to the Token-Jaccard similarity score. All pairs from the top of the list (including only instances that occur at most once) are selected as positive training examples. The negative training examples are generated by permuting the pairs selected as positive examples. The selected training examples are applied in the property alignment (attribute matching) step, using column similarities of attributes over the matching and non-matching examples. Following this, the training examples are converted into feature vectors. The two sets of feature vectors

are then applied for training the classification model. It has been demonstrated in the experimental evaluation that the proposed approach, with SVM as a learning method, was competitive with the supervised SVM on some data sets.

In [24], an approach based on combination of ensemble learning and automatic self-learning was proposed. Ensemble learning is a process of training and combining a number of different classification models in order to improve the overall performance. The initial training set is generated following the similar idea as in [6]. To improve the quality of the training set, they incorporate unsupervised attribute weighting into the process of automatic selection of seeds. The authors propose to first create multiple views of the record pairs by generating the comparison vectors applying different matching schemes. Each matching scheme is composed of different combinations of attributes and similarity measures. Following this, a self-learning process is performed with each of the views individually. At the end, the final classification decisions are made based on the combined predictions of all individual models. In order to improve the final classifier ensemble model, two unsupervised diversity measures are proposed; first one for selecting an optimal combination of views and second one for selecting subset of the most diverse self-learning models. Finally, they use the contribution ratio of the selected self-learning models to eliminate those with poor accuracy. Application of multi-view ensemble learning improves the learning process and allows to address the problem of selecting the most appropriate matching schemes without labelled data. However, application of ensemble learning causes increased number of record comparisons performed in the linking process, which may not scale with larger data sets given.

Approaches that generate their own training data have a clear advantage over techniques such as AL, since they do not require any data to be labelled manually. However, they rely on the selection of high-quality seeds, which is not trivial. Most of them apply a similarity threshold to decide whether a pair of records is similar/dissimilar enough to be included in the initial training set. Selection of an appropriate threshold may require a deep knowledge of the data and the domain.

3.3.4 Objective Function Optimisation

In recent work, unsupervised techniques for RL based on maximising the value of a pseudo F-measure [34] were proposed [33, 34]. Pseudo F-measure is formulated based on the assumption that while different records often represent the same entity in different repositories, distinct record within one data set is expected to denote distinct entity. The main advantage of this measure is the fact that it can be calculated using only unlabelled records.

For two sets of records, S and T, and a list of record pairs $M \in S \times T$ selected by an algorithm as matches, the pseudo precision and pseudo recall are defined as follows:

$$P(M) = \frac{|s|\exists t : (s,t) \in M|}{\sum_s |t : (s,t) \in M|} \tag{3.5}$$

$$R(M) = \frac{|s|\exists t : (s,t) \in M| + |t|\exists s : (s,t) \in M|}{\sum_s |S| + |T|} \tag{3.6}$$

Following this, the pseudo F-measure is formulated as:

$$F(M) = (1+\beta^2)\frac{P(M)R(M)}{\beta^2 P(M) + R(M)} \tag{3.7}$$

The idea is to find the decision rule for record matching, which maximises the value of the objective function (pseudo F-measure) applying genetic programming [34] or a hierarchical grid search [33]. In both cases, a search space of RL rules is formulated by manipulating weights and similarity measures for pairs of attributes, modifying the similarity threshold values and changing the aggregation function for individual similarities. Applying the pseudo F-measure to evaluate the linkage performance allows to eliminate the requirement for labelled data, and it also addresses the problem of selecting an optimal similarity measure. However, a comparative analysis shows that there is a considerable gap between their performance and that of supervised systems. Apart from this, a recent study demonstrated that on real data the pseudo F-measure is often not correlated with the true F-measure, which raises concerns about whether it can be applied to predict the real accuracy of an RL model [33]. Finally, the method makes some assumptions regarding the data set, which reduces its generality.

A novel optimisation approach to unsupervised RL is proposed in [23]. In this work, a scoring technique using a weighted sum formulation is defined for aggregating similarities between records. Considering a pair of records, the overall similarity is measured as a weighted sum of similarities along each of the attributes and using each of the available similarity measures. The goal of the method is in learning the set of weights so that the aggregated similarity score estimate the linkage likelihood effectively. Following this, an objective function is defined, which is motivated by the unambiguity of the scoring. In other words, the aim is to estimate the weights so that the result's similarity scoring for each pair of records is either closer to the lower extreme (match) or closer to the higher extreme (non-match). A gradient decent algorithm is adopted for optimising the objective function. Following the optimisation process, all the record pairs are ranked in decreasing order according to their similarity scores. The method does not require any labelled data, and it addresses the problem of similarity measure selection. Given the nature of gradient decent algorithm, it is also more efficient when applied with large data sets than some of the other existing approaches. However, for the method to be applicable in a practical scenario, one would need to apply a cut-off point in the ranked list, so that record pairs above the cut-off could be regarded as match, and those below may be considered as non-match. Estimation of the cut-off

point can be problematic without gold-standard labelled data, which narrows the applicability of the proposed approach.

3.3.5 Knowledge Graph Embedding

Recently, great attention was given to learning embeddings of entities and relations [4] in knowledge graphs to then use them for entity linking. A typical knowledge graph is represented by triple facts of $(head, label, tail)$ or (h, l, t), which indicate that there exists a relationship of type $label$ between the entities $head$ and $tail$. A knowledge graph is represented as $KG = (E, R, T)$ where E, R, T refer to entities, relations and triples, respectively.

In [4], an energy-based model is used for learning low-dimensional embeddings of entities and relations such that $h + r \simeq t$. The aim is to find embeddings such that, for two entities h, t and a relationship l, if (h, l, t) holds then the embedding of t should be close to the embedding of t plus a vector that depends on l. The energy function is defined as:

$$d(h, l, t) = \|h + r - t\|_{L_1/L_2} \tag{3.8}$$

For a given set S of triplets, the embeddings are learnt via minimisation of the objective function:

$$L = \sum_{(h,l,t)\in S} \sum_{(h',l',t')\in S'} [\gamma + d(h, l, t) - d(h', l', t')]_+ \tag{3.9}$$

where $[x]_+ = max\{0, x\}$, and S' stands for negative sample set of S:

$$S' = (h', l, t)|h' \in E \cup (h, l, t')|t' \in E \tag{3.10}$$

In later work, a number of methods which can deal with more complicated relations were proposed [22, 29, 45]. In [52], a model was proposed which encodes the embeddings of entities and relationships following [4] and [44] for each of the multiple knowledge graphs. Following this, the knowledge embeddings from different knowledge graphs are joined into unified semantic space using a small seed set of linked entities. The knowledge-based embeddings and joint embeddings are applied for linking entities from different knowledge graphs.

In [13], a self-learning algorithm was applied together with knowledge graph embedding for unsupervised entity linking. The proposed model does not require any prior linked entities. However, it assumes that the alignments between attributes across the knowledge graphs are given. The proposed framework consists of two components: knowledge graph embeddings and entity linking. The self-learning is applied as a feedback mechanism from the entity linking to the graph knowledge embeddings. In the first iteration, the knowledge embeddings are initialised

randomly. Following this, the entity pairs are sorted according to their semantic distance. The top entities are selected and labelled as matches. In the next iterations, the learnt triples are used to update all the embeddings. The model was evaluated on some of the RL data sets; however, it was not compared against any other type of methods such as learning-based methods.

3.4 Limitations and Future Research Directions

The problem of semi-supervised and unsupervised RL has been addressed in a vast number of studies in the past decade. The majority of the work in this area has been focused on relaxing the assumptions on existence of a training data set available for learning the linkage model. Following the aforementioned review, one can note that significant progress was made in this area. The existing methods allow to reduce the manual effort of labelling data or entirely eliminate it. However, the existing approaches make some assumptions regarding the data, which limits their application in real-world scenarios. Most of the existing work focuses on a scenario where the data to be linked is structured and consists of records represented by a common set of well-aligned attributes. Some RL methods were developed to deal with data stored in the RDF format. Only a few of the aforementioned papers address the problem of schema matching. Most of the work assumes that the pairs of records' attributes to be used for comparison are given. The problem of selecting optimal similarity measures has only been addressed in a few of the discussed papers.

Below, we list some topics that have not been addressed by any of the discussed work on semi-supervised and unsupervised RL. The challenge related to those topics rises in unsupervised scenarios, when no ground truth information is provided regarding the matching status of the data to be linked.

- *Unstructured and complex data.* Nowadays, increasingly data is stored in free text format, where no specific attributes are provided, for example, social media posts, news article, emails and Web pages. With this type of data, preprocessing and data extraction techniques need to be applied before the RL process can be performed.
- *Heterogeneous data types.* Different data sources, even within the same domain, can be extremely heterogeneous. For example, medical records of a patient can be stored in many different formats, such as images, structured record in a database or reports in a free text. In order to link such multi-type data, methods for comparing and determining similarity between data in different formats are required.
- *Linking accuracy.* Many different techniques of reducing or eliminating manually labelled data from the learning process have been proposed. However, there is still a gap between quality of the linking of the unsupervised and supervised RL methods. Most of the aforementioned methods are based on similarity values obtained from similarity measures and in some cases, additional relational

information. This information have been proven to be very effective while linking records with supervised RL techniques. Nevertheless, there is a need for some more advanced approaches that can provide high linking quality when no labelled data or domain expertise is provided.

3.5 Conclusions

In this paper, we provided an overview of the most recent advances in the unsupervised and semi-supervised approaches to identifying pairs of records referring to the same entity. In particular, we considered different techniques that have been explored in order to relax the assumption on manually labelled data being available for learning RL models. We first provided an introduction and talked about the significance of RL as a part of a bigger data integration problem. We described the RL process in more detail and provided a formal definition of the problem. We also gave a list of data sets that are commonly applied for evaluation of different RL models. We then looked in detail on two families of approaches, namely semi-supervised and unsupervised. With semi-supervised approaches, we considered crowdsource-based methods and learning-based techniques. Within the family of unsupervised approaches, we covered graph-based models, learning-based models, models based on an optimisation of an objective function and also knowledge graph embedding models.

The focus of this chapter was to provide information on any of the recent techniques that have been explored to address the problem of semi-supervised and unsupervised RL. The chapter provides a good overview of the recent advances and points out some remaining challenges and directions for future research. This chapter should also be helpful for readers who are looking to decide about the most appropriate RL model to use with their data.

References

1. Arasu, A., Gotz, M., Kaushik, R.: On active learning of record matching packages. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data, pp. 783–794. ACM, New York (2010)
2. Baxter, R., Christen, P., Churches, T., et al.: A comparison of fast blocking methods for record linkage. In: ACM SIGKDD, vol. 3, pp. 25–27. Citeseer (2003)
3. Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., Fienberg, S.: Adaptive name matching in information integration. IEEE Intell. Syst. **18**(5), 16–23 (2003)
4. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in Neural Information Processing Systems, pp. 2787–2795 (2013)
5. Chai, C., Li, G., Li, J., Deng, D., Feng, J.: Cost-effective crowdsourced entity resolution: a partial-order approach. In: Proceedings of the 2016 International Conference on Management of Data, pp. 969–984. ACM, New York (2016)

6. Christen, P.: Automatic record linkage using seeded nearest neighbour and support vector machine classification. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 151–159. ACM, New York (2008)
7. Christen, P.: Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer, Berlin (2012)
8. Christen, P.: A survey of indexing techniques for scalable record linkage and deduplication. IEEE Trans. Knowl. Data Eng. **24**(9), 1537–1555 (2012)
9. Cohen, W., Ravikumar, P., Fienberg, S.: A comparison of string metrics for matching names and records. In: KDD Workshop on Data Cleaning and Object Consolidation, vol. 3, pp. 73–78 (2003)
10. DuVall, S.L., Kerber, R.A., Thomas, A.: Extending the Fellegi–Sunter probabilistic record linkage method for approximate field comparators. J. Biomed. Inform. **43**(1), 24–30 (2010)
11. Elfeky, M.G., Verykios, V.S., Elmagarmid, A.K.: Tailor: a record linkage toolbox. In: Proceedings 18th International Conference on Data Engineering, 2002, pp. 17–28. IEEE, Piscataway (2002)
12. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: a survey. IEEE Trans. Knowl. Data Eng. **19**(1), 1–16 (2007)
13. Guan, S., Jin, X., Jia, Y., Wang, Y., Shen, H., Cheng, X.: Self-learning and embedding based entity alignment. In: 2017 IEEE International Conference on Big Knowledge (ICBK), pp. 33–40. IEEE, Piscataway (2017)
14. Herzog, T.N., Scheuren, F.J., Winkler, W.E.: Data Quality and Record Linkage Techniques. Springer, Berlin (2007)
15. Isele, R., Bizer, C.: Learning expressive linkage rules using genetic programming. Proc. VLDB Endowment **5**(11), 1638–1649 (2012)
16. Isele, R., Bizer, C.: Active learning of expressive linkage rules using genetic programming. Web Semant. Sci. Serv. Agents World Wide Web **23**, 2–15 (2013)
17. Iwata, T., Ishiguro, K.: Robust unsupervised cluster matching for network data. Data Min. Knowl. Disc. **31**(4), 1132–1154 (2017)
18. Iwata, T., Lloyd, J.R., Ghahramani, Z.: Unsupervised many-to-many object matching for relational data. IEEE Trans. Pattern Anal. Mach. Intell. **38**(3), 607–617 (2016)
19. Jaccard, P.: Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines. Bull. Soc. Vaud. Sci. Nat. **37**, 241–272 (1901)
20. Jain, S., Neal, R.M.: A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. J. Comput. Graph. Stat. **13**(1), 158–182 (2004)
21. Jaro, M.A.: Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. J. Am. Stat. Assoc. **84**(406), 414–420 (1989)
22. Jia, Y., Wang, Y., Lin, H., Jin, X., Cheng, X.: Locally adaptive translation for knowledge graph embedding. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, pp. 992–998 (2016)
23. Jurek, A., Deepak, P.: It pays to be certain: unsupervised record linkage via ambiguity minimization. In: Proceedings of 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (2018)
24. Jurek, A., Hong, J., Chi, Y., Liu, W.: A novel ensemble learning approach to unsupervised record linkage. Inf. Syst. **71**, 40–54 (2017)
25. Kejriwal, M., Miranker, D.P.: Semi-supervised instance matching using boosted classifiers. In: European Semantic Web Conference, pp. 388–402. Springer, Berlin (2015)
26. Kejriwal, M., Miranker, D.P.: An unsupervised instance matcher for schema-free RDF data. Web Semant. Sci. Serv. Agents World Wide Web **35**, 102–123 (2015)
27. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. Sov. Phys. Dokl. **10**, 707–710 (1966)
28. Li, G.: Human-in-the-loop data integration. Proc. VLDB Endowment **10**(12), 2006–2017 (2017)

29. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI Conference on Artificial Intelligence, vol. 15, pp. 2181–2187 (2015)
30. Marcus, A., Wu, E., Karger, D., Madden, S., Miller, R.: Human-powered sorts and joins. Proc. VLDB Endowment **5**(1), 13–24 (2011)
31. Naumann, F., Herschel, M.: An introduction to duplicate detection. Synth. Lect. Data Manage. **2**(1), 1–87 (2010)
32. Ngomo, A.C.N., Lyko, K.: Eagle: efficient active learning of link specifications using genetic programming. In: Extended Semantic Web Conference, pp. 149–163. Springer, Berlin (2012)
33. Ngomo, A.C.N., Lyko, K.: Unsupervised learning of link specifications: deterministic vs. non-deterministic. In: Proceedings of the 8th International Conference on Ontology Matching, vol. 1111, pp. 25–36 (2013). CEUR-WS.org
34. Nikolov, A., d'Aquin, M., Motta, E.: Unsupervised learning of link discovery configuration. In: Extended Semantic Web Conference, pp. 119–133. Springer, Berlin (2012)
35. Sadinle, M., et al.: Detecting duplicates in a homicide registry using a Bayesian partitioning approach. Ann. Appl. Stat. **8**(4), 2404–2434 (2014)
36. Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE Mobile Comput. Commun. Rev. **5**(1), 3–55 (2001)
37. Sherif, M.A., Ngomo, A.C.N., Lehmann, J.: W ombat–a generalization approach for automatic link discovery. In: European Semantic Web Conference, pp. 103–119. Springer, Berlin (2017)
38. Steorts, R., Hall, R., Fienberg, S.: Smered: a Bayesian approach to graphical record linkage and de-duplication. In: Artificial Intelligence and Statistics, pp. 922–930 (2014)
39. Steorts, R.C., Ventura, S.L., Sadinle, M., Fienberg, S.E.: A comparison of blocking methods for record linkage. In: International Conference on Privacy in Statistical Databases, pp. 253–268. Springer, Berlin (2014)
40. Steorts, R.C., et al.: Entity resolution with empirically motivated priors. Bayesian Anal. **10**(4), 849–875 (2015)
41. Steorts, R.C., Hall, R., Fienberg, S.E.: A Bayesian approach to graphical record linkage and deduplication. J. Am. Stat. Assoc. **111**(516), 1660–1672 (2016)
42. Wang, J., Li, G., Yu, J.X., Feng, J.: Entity matching: how similar is similar. Proc. VLDB Endowment **4**(10), 622–633 (2011)
43. Wang, J., Li, G., Kraska, T., Franklin, M.J., Feng, J.: Leveraging transitive relations for crowdsourced joins. In: Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, pp. 229–240. ACM, New York (2013)
44. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph and text jointly embedding. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1591–1601 (2014)
45. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI Conference on Artificial Intelligence, vol. 14, pp. 1112–1119 (2014)
46. Wang, Q., Vatsalan, D., Christen, P.: Efficient interactive training selection for large-scale entity resolution. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 562–573. Springer, Berlin (2015)
47. Wang, S., Xiao, X., Lee, C.H.: Crowd-based deduplication: an adaptive approach. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 1263–1277. ACM, New York (2015)
48. Winkler, W.E.: String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In: American Statistical Association 1990 Proceedings of the Section on Survey Research Methods, pp. 354–359 (1990)
49. Zhang, D., Guo, L., He, X., Shao, J., Wu, S., Shen, H.T.: A graph-theoretic fusion framework for unsupervised entity resolution. In: Proceedings of the 34th IEEE International Conference on Data Engineering (2018)
50. Zheng, Y., Li, G., Li, Y., Shan, C., Cheng, R.: Truth inference in crowdsourcing: is the problem solved? Proce. VLDB Endowment **10**(5), 541–552 (2017)

51. Zhu, L., Ghasemi-Gol, M., Szekely, P., Galstyan, A., Knoblock, C.A.: Unsupervised entity resolution on multi-type graphs. In: International Semantic Web Conference, pp. 649–667. Springer, Berlin (2016)
52. Zhu, H., Xie, R., Liu, Z., Sun, M.: Iterative entity alignment via joint knowledge embeddings. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, pp. 4258–4264. AAAI Press, Menlo Park (2017)

Chapter 4
A Review of Unsupervised and Semi-supervised Blocking Methods for Record Linkage

Kevin O'Hare, Anna Jurek-Loughrey, and Cassio de Campos

Abstract Record linkage, referred to also as entity resolution, is a process of identifying records representing the same real-world entity (e.g. a person) across varied data sources. To reduce the computational complexity associated with record comparisons, a task referred to as blocking is commonly performed prior to the linkage process. The blocking task involves partitioning records into blocks of records and treating records from different blocks as not related to the same entity. Following this, record linkage methods are applied within each block significantly reducing the number of record comparisons. Most of the existing blocking techniques require some degree of parameter selection in order to optimise the performance for a particular dataset (e.g. attributes and blocking functions used for splitting records into blocks). Optimal parameters can be selected manually but this is expensive in terms of time and cost and assumes a domain expert to be available. Automatic supervised blocking techniques have been proposed; however, they require a set of labelled data in which the matching status of each record is known. In the majority of real-world scenarios, we do not have any information regarding the matching status of records obtained from multiple sources. Therefore, there is a demand for blocking techniques that sufficiently reduce the number of record comparisons with little to no human input or labelled data required. Given the importance of the problem, recent research efforts have seen the development of novel unsupervised and semi-supervised blocking techniques. In this chapter, we review existing blocking techniques and discuss their advantages and disadvantages. We detail other research areas that have recently arose and discuss other unresolved issues that are still to be addressed.

K. O'Hare (✉) · A. Jurek-Loughrey
School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast, UK
e-mail: kohare08@qub.ac.uk; a.jurek@qub.ac.uk

C. de Campos
Buys Ballotgebouw, Utrecht University, Utrecht, The Netherlands
e-mail: c.decampos@uu.nl

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_4

4.1 Record Linkage

Record Linkage (RL) is a process of identifying and linking pairs of records from multiple data sources (e.g. databases) representing the same real-world entity (referred to as matching records). An overview of a general RL process is demonstrated in Fig. 4.1.

Records are first standardised in order to remove inconsistencies between otherwise matching values. This may consist of removing non-alphanumeric characters or replacing multiple spelling variations of the same word with a single common spelling (e.g. "$John$", "$Johnny$", "Jon" $\rightarrow$ "$John$"). Following this, the standardised records are divided into groups (blocks) in such a manner that records within each group have a high chance of being linked in the subsequent linkage process. This is necessary to reduce the computational cost of linkage as record pair comparison is computationally expensive and grows in number exponentially with dataset sizes. Figure 4.2 depicts a simple example of a blocking process in which records are placed into blocks according to their *Surname* attribute value. In this case, the blocking process reduces the comparison space from 45 comparisons $\left(\text{i.e. } \frac{n(n-1)}{2} \text{ where } n = \text{number of records}\right)$ to only 8 comparisons, 2 of which are of matching record pairs (highlighted in green).

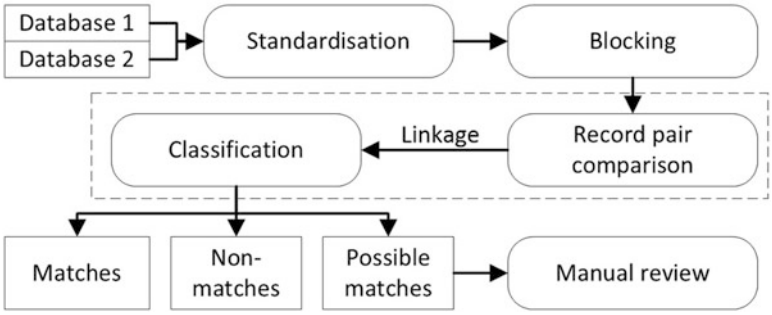


Fig. 4.1 General overview of RL process

	First Name	Surname	Postcode	Age
R1	Paul	Smyth	BT5 6AE	32
R2	Robert	Paulson	BT4 3DC	56
R3	Susan	Rivers	BT4 3DC	62
R4	Joanne	Rivers	BT2 3RD	84
R5	Neil	Smith	BT6 7ED	51
R6	Paul	Smith	BT6 5AE	32
R7	Bob	Paulson	BT4 3DC	56
R8	Joan	Rivers	BT2 3RD	83
R9	Joan	Smith	BT2 3RD	83
R10	Adam	Smyth	BT4 7AR	29

"Smyth"

R1	Paul	Smyth	BT5 6AE	32
R10	Adam	Smyth	BT4 7AR	29

"Paulson"

R2	Robert	Paulson	BT4 3DC	56
R7	Bob	Paulson	BT4 3DC	56

"Rivers"

R3	Susan	Rivers	BT4 3DC	62
R4	Joanne	Rivers	BT2 3RD	84
R8	Joan	Rivers	BT2 3RD	83

"Smith"

R5	Neil	Smith	BT6 7ED	51
R6	Paul	Smith	BT6 5AE	32
R9	Joan	Smith	BT2 3RD	83

Fig. 4.2 Example of a simple blocking scheme

Following blocking, linkage is performed exclusively on the record pairs within the same block. The blocked record pairs are compared and assigned a similarity value which informs whether they should be considered as a match, non-match or possible match requiring further manual review.

A good blocking method places as many matching record pairs, and as few non-matching record pairs into the same blocks allowing for an efficient subsequent linkage phase. Blocking methods are commonly evaluated with labelled data (with known matching status of each record pair) using evaluation metrics such as *Reduction Ratio* (RR), *Pairs Completeness* (PC) and a harmonic mean $F_{RR,PC}$ of RR and PC [21].

Definition 1 (Reduction Ratio) For two datasets, A and B, Reduction Ratio is defined as:

$$RR = 1 - \frac{N}{|A| \times |B|}, \tag{4.1}$$

where $|A| \times |B|$ is the total number of unique possible among both datasets, and $N \leq (|A| \times |B|)$ is the number of record pairs formed by a blocking method.

RR indicates how much the comparison space is reduced after the blocking phase. In the example from Fig. 4.2, the potential comparison space is reduced from 45 record pairs to 8, which equates to $RR = 1 - \frac{8}{45} = 0.8\dot{2}$.

Definition 2 (Pairs Completeness) Pairs Completeness is defined as:

$$PC = \frac{N_m}{|M|}, \tag{4.2}$$

where $N_m \leq |M|$ is the number of matching record pairs contained within the reduced comparison space after blocking, and $|M|$ is the number of matches within the entire dataset.

PC is the ratio of matching record pairs found within the formed blocks. In Fig. 4.2, there are five record pairs that are considered matches (i.e. $\{R_1, R_6\}$, $\{R_2, R_7\}$, $\{R_4, R_8\}$, $\{R_4, R_9\}$ and $\{R_8, R_9\}$) which equates to $PC = \frac{2}{5} = 0.4$.

One can notice that there is a trade-off between RR and PC. Comparing all record pairs (placing all the records in the same block) minimises RR but maximises PC, whereas performing no comparisons at all (placing each record in an individual block) maximises RR and minimises PC. Ideally, one looks for a blocking method that achieves an optimal degree of both RR and PC. A commonly applied evaluation metric, which balances the trade-off between RR and PC, is the harmonic mean of RR and PC.

Definition 3 (Harmonic Mean of RR and PC) For a given RR and PC, the harmonic mean is defined as:

$$F_{RR,PC} = \frac{2 \cdot RR \cdot PC}{RR + PC}. \tag{4.3}$$

A perfect blocking method optimises both RR and PC so that only matching record pairs are compared. To perfectly partition record pairs in such a manner is a non-trivial task. Many different automatic blocking methods have been proposed in order to address it.

Supervised approaches use labelled training data to generate highly proficient blocking methods. However, in the majority of real-world scenarios we do not have any information regarding the matching status of records obtained from multiple sources. Therefore, there is a significant demand for blocking approaches that can achieve good proficiency with little or no need for labelled data.

In the following sections, we review existing unsupervised and semi-supervised blocking methods discussing their advantages and disadvantages.

4.2 Blocking Approaches

In this section, we discuss different categories of blocking algorithms, providing examples and details on how they work as well as what advantages and disadvantages they have.

4.2.1 Standard Key-Based Blocking

Standard blocking uses a set of blocking keys to determine which records should be placed in the same block. Consider a dataset of records $R = r_1, \ldots, r_n$ where each record is represented by attributes from a scheme $A = a_1, \ldots, a_m$. Accordingly, we can represent a record r_i as $[r_{i,1}, \ldots, r_{i,m}]$, where $r_{i,j}$ is the value that the ith record takes for the jth attribute. A blocking key is defined as follows:

Definition 4 (Blocking Key) A blocking key is an $\langle a_j, h \rangle$ combination where $a_j \in A$ is an attribute and h is an indexing function. For each $r_i \in R$, h takes $r_{i,j}$ as an input and provides a set of values, referred to as blocking key values (BKV), as an output.

BKVs determine into which block(s) records are placed. Each unique BKV refers to a specific block. Due to the complexity of datasets (which may contain a variety of issues such as missing values, typographical errors, acronyms or initialisations), a single blocking key is rarely likely to capture all matching record pairs efficiently. Multiple blocking keys therefore tend to be used in the form of a blocking scheme.

Definition 5 (Blocking Schemes) Given a set of individual blocking keys, $K = k_1, \ldots, k_{k'}$, a blocking scheme is a combination of keys that are used collectively.

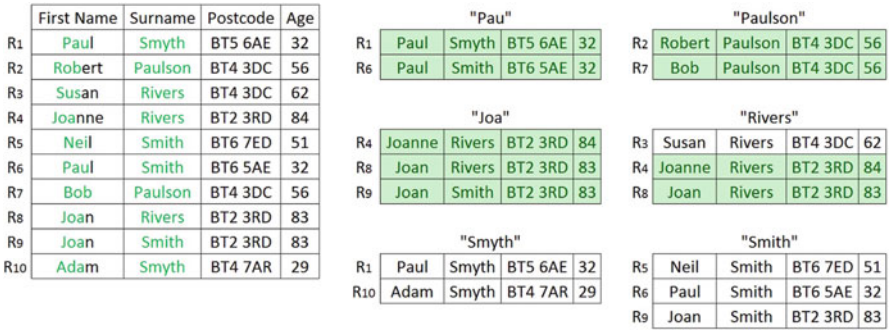


Fig. 4.3 Example of a disjunctive blocking scheme. For clarity, only those blocks that contain more than one record, and thus form at least one record pair, are presented

This combination can be disjunctive, $\langle k_i \rangle \cup \ldots \cup \langle k_j \rangle$, conjunctive, $\langle k_i \rangle \cap \ldots \cap \langle k_j \rangle$, or a disjunction of conjunctions (Disjunctive Normal Form, DNF), $\langle \langle k_i \rangle \cap \ldots \cap \langle k_j \rangle \rangle \cup \ldots \cup \langle \langle k_{i'} \rangle \cap \ldots \cap \langle k_{j'} \rangle \rangle$.

In Fig. 4.3, a disjunctive blocking scheme is applied (i.e. $\langle$*First Name, 1st 3 characters*$\rangle \cup \langle$*Surname, Exact Match*$\rangle$). This disjunctive blocking scheme captures more of the matching record pairs (highlighted in green) than that of Fig. 4.2 but at a cost of generating additional record comparisons.

Effective blocking schemes may be created manually by a domain expert [13, 16] or can be automatically learned using a blocking scheme learning (BSL) algorithm [6, 34, 47]. BSL algorithms tend to follow a common general approach in which individual blocking keys are first evaluated and then ranked according to a criteria (e.g. ratio of matches to non-matches). The top-ranked individual keys continue to form conjunctions until a satisfaction criteria is met (e.g. maximum number of record pairs formed, and maximum conjunction length). The top-ranked keys and conjunctions are then selected as a blocking scheme.

BSL algorithms have been demonstrated to be very effective [6, 34, 47] but often require labelled data in order to indicate the "best" keys and conjunctions in a supervised manner [6, 47]. The problem of labelled data requirement for evaluating blocking keys was addressed in [34]. In this work, an unsupervised BSL approach, which automatically labels its own data from the target dataset, was proposed. Records are first placed into blocks according to their commonly shared tokens (i.e. individual words). Following this, a window of predetermined size is slid over the records within each block. Record pairs within the window are then compared using the (log) Term Frequency-Inverse Document Frequency (TF-IDF) measure (Eq. (4.4)). The Log TF-IDF measure is formally defined as:

$$sim(t_1, t_2) = \sum_{q \in t_1 \cap t_2} w(t_1, q) \cdot w(t_2, q), \tag{4.4}$$

where

$$w(t,q) = \frac{w'(t,q)}{\sqrt{\sum_{q \in t} w'(t,q)^2}}, \tag{4.5}$$

and

$$w'(t,q) = \log(tf_{t,q} + 1) \cdot \log\left(\frac{|R|}{df_q} + 1\right) \tag{4.6}$$

where (t_1, t_2) represents a record pair, $w(t,q)$ is the normalised TF-IDF weight of a term q in a record t, $tf_{t,q}$ represents the term frequency of q in t, $|R|$ is the total number of records in the dataset R and df_q is the document frequency of the term q in the cohort. Predefined quantities of the most and least similar record pairs are labelled as positives and negatives, respectively. All blocking keys are applied to the labelled record pairs creating binary feature vectors (i.e. 1 or 0 if the record pair agree or disagree by the respective key). Keys that cover above a predetermined proportion of negative vectors are omitted from further consideration. If the disjunction of all permitted keys is unable to cover a predetermined proportion of the positive vectors, then a message is returned explaining that a blocking scheme cannot be learned under the current parameters. Otherwise, the keys are ranked by Fisher Score [15] and applied in descending order until the predetermined proportion of positive vectors is covered. Keys that cover new positive vectors are iteratively added to the blocking scheme. For DNF schemes, a similar approach is taken except the list of permitted keys is supplemented by conjunctions of keys restricted to a predetermined maximum conjunction length. Furthermore, only those conjunctions that have Fisher Score equal to or greater than the average Fisher Score of the permitted keys are added. The authors note that their work is the only one to cast blocking scheme learning as a feature selection problem. Kejriwal and Miranker [34] evaluate their unsupervised BSL approach against the supervised baseline approach of Bilenko et al. [6] using RR, PC and $F_{RR,PC}$. In [6], blocking keys that cover too many negatives pairs, and negative pairs covered by too many blocking keys are excluded from consideration. The remaining keys (and their conjunctions if applicable) are then ranked by their ratio of covered positives to covered negatives and applied iteratively as either a disjunctive or DNF blocking scheme. In [34], experimental evaluations indicate that their novel unsupervised BSL algorithm is seen to at least equal the performance of the supervised baseline in most cases. The authors document great success for both the automatic labelling and BSL approach for each dataset but it is worth noting that these experiments were only carried out with a small number of small datasets, the largest of which only containing 1295 records. Whether this holds for substantially larger datasets of higher dimensions is unknown. A further potential issue is that the automatic labelling approach requires comparing all record pairs of a dataset that share any

common tokens (i.e. individual words). For substantially large datasets of high dimension, this may be non-trivial to carry out.

q-Gram indexing [4, 9, 54, 56, 69] is a popular older unsupervised blocking approach that extends upon standard blocking. It is commonly used as a benchmark for other approaches to compare against as it is easy to comprehend and implement. It was first generalised in [20] and experimentally evaluated using Bi-Grams (i.e. $q = 2$) in [10]. The basic concept is that a defined blocking key is used to generate BKVs for different records of a dataset, as per the standard blocking approach, which are then split into lists of q-Grams (i.e. substrings of length q). Permutations of sub-lists are then constructed according to a threshold value from $0.0 \rightarrow 1.0$. The threshold value is multiplied by the length of the entire q-Gram list, and the rounded answer dictates the length of the permutations of q-Gram sub-lists to be constructed. The different permutations of sub-lists are then concatenated to form different blocking keys. Inverted index blocks are then created for each key with the records placed accordingly. Figure 4.4 depicts Bi-Gram indexing being applied to the same records of Fig. 4.2. In this case, the DNF blocking key $\langle$*First Name, 1st 3 characters*$\rangle \cap \langle$*Postcode, Last 3 characters*$\rangle$ is used to generate the individual BKVs. These BKVs are then split into Bi-Gram lists, and a threshold of 0.6 is used to further generate the sub-lists. As each Bi-Gram list consists of 5 Bi-Grams, this results in $5 \times 0.6 = 3$ sub-lists of Bi-grams per record. In this case, the comparison space is reduced to only four unique record pair comparisons, all of which are of matching record pairs (i.e. $\{R_4, R_8\}$, $\{R_4, R_9\}$, $\{R_8, R_9\}$ and $\{R_2, R_7\}$). q-Gram indexing of a dataset results in a maximum of $O(\frac{n^2}{b})$ comparisons where b is the number of indices or blocks created. As such, the efficiency and complexity

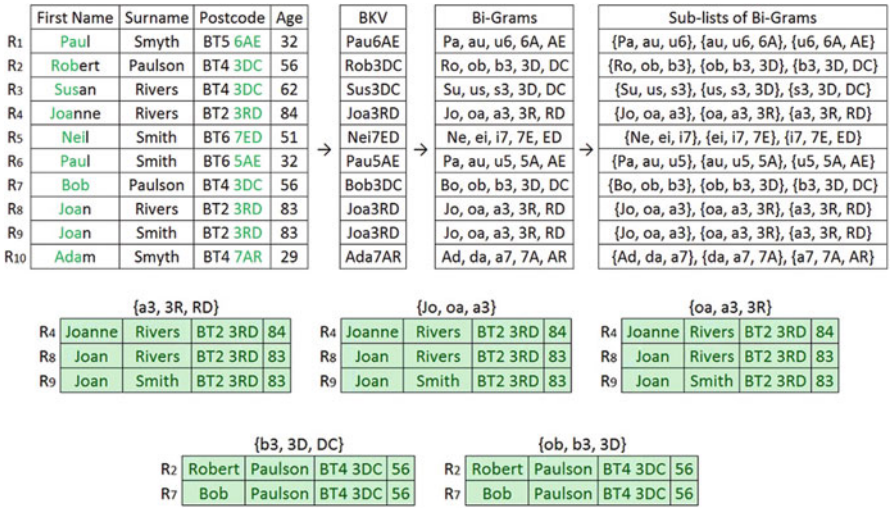


Fig. 4.4 Example of q-Gram indexing where $q = 2$ (i.e. Bi-Gram indexing). For clarity, only those blocks that contain more than one record, and thus form at least one record pair, are presented

of q-Grams indexing is very much dependent upon the value of q used. Smaller q values tend to better accommodate typographical errors between near-similar BKVs of matching records but at a cost of generating more indices which adds to the complexity. Conversely, larger values of q reduce complexity in that fewer indices are created but at a potential cost to PC. Additionally, the q-Gram sub-list generation time tends to be dominated by the recursive generation of sub-lists for longer BKVs meaning that BKV length should also be considered when selecting q [9]. By allowing records to be inserted into multiple blocks, q-Gram indexing allows for fuzzy-matching which is a desirable quality in a blocking technique.

In [4], Bi-Gram indexing is shown to be superior to the standard blocking and sorted neighbour (discussed in Sect. 4.2.3) approaches. However, in [26] q-Gram indexing was compared to a number of other blocking approaches and was found to be among the worst performing in terms of computational runtime. In [9], it is shown that it is possible to estimate the overhead cost of q-Gram indexing. In the same paper, the authors note that although q-Gram indexing tends to have higher PC than standard blocking and sorted neighbour approaches, it also has poorer RR leading to a more time-consuming linkage process. In [63], the authors state that weighting every Bi-Gram equally negatively affects performance. In [54, 56], an extended variation of q-Gram indexing is described which according to the authors improves the standard variation by generating indexing keys with higher discriminability. In doing so, they attempt to detect all of the duplicates that standard q-Gram indexing would detect but in fewer comparisons. They achieve this by concatenating different combinations of at least $L = max(1, [k \cdot T])$ sequential q-Grams of a BKV where L is derived from user-defined parameters $T \in [0, 1]$ and k, which is the number of q-Grams in the original BKV.

Another extension to standard blocking is that of Suffix-Array-based indexing [1, 7, 54, 56] in which records are blocked according to a list of suffixes of minimum length l_m derived from the respective BKV of a record by a defined blocking key. As one can imagine, the effectiveness of Suffix-Array-based indexing is dependent upon the set value of l_m. Smaller l_m values result in larger lists of suffix values meaning that each record is placed in more blocks, improving PC at a cost to RR. On the other hand, large l_m values result in shorter lists of more distinct suffix values, improving RR but at a potential negative cost to PC. If l_m is equal to the size of the BKV itself, then the approach is no different than standard blocking. Very common suffixes are often excluded from consideration as their inclusion results in extremely large blocks. In [7], Suffix-Array-based indexing is shown to have some of the lowest PC results among a number of blocking approaches for a number of synthetically generated evaluation datasets. An extended variation of Suffix-Array-based indexing is defined in [54, 56] in which all substrings of length l_m of a BKV are considered rather than just the suffixes. In [54], variations of both Suffix-Array-based indexing and its extended variation are defined in which all tokens of all values of all entities are used rather than BKVs generated from a defined blocking key.

4.2.2 Schema-Agnostic Blocking Approaches

A schema provides information regarding the attributes of a dataset (e.g. the number of attributes, column names or data types). Often, prior knowledge of a dataset schema is necessary in order to select and use the most discriminative attributes that are resistant to error or missing values and in cases of heterogeneous data sources are common to each dataset. With schema-agnostic blocking methods, no such prior information is needed. In this subsection, we detail commonly used schema-agnostic blocking approaches as well as some variations that improve upon their original implementation.

A commonly used schema-agnostic blocking approach is *Token-Blocking* [34, 36, 37, 51, 52] in which individual records are split into bags of all possible tokens (individual words) and grouped by their commonly shared tokens. Figure 4.5 depicts *Token-Blocking* applied to the same records of Fig. 4.2. *Token-Blocking* is favoured by many for a number of reasons. The basic concept is very easy to comprehend and implement and is applicable to both homogeneous and heterogeneous data sources. Matching record pairs are highly likely to share at least one common token therefore this approach tends to result in high PC in all datasets. A downside of *Token-Blocking* is that many non-matching record pairs may also contain at least one common token, which results in low RR. A variation of *Token-Blocking* improves upon this by grouping records by common token sequences of length n rather than individual tokens. This improves RR significantly but at a potential negative cost to PC. A recent research trend referred to as Meta-blocking (discussed in Sect. 4.2.6) has been used to improve the RR of *Token-Blocking* with little to no negative effect to PC.

Token-Blocking can be further improved by stipulating that the common tokens between records must be present in the same attribute column(s) in order for the records to be placed in the same block. For homogeneous cases, this is easy to implement as every record of a single dataset inherently shares the same schema. For heterogeneous cases, only those attributes that are common to each of the multiple data sources may be used. With no knowledge about the schemas, it can be difficult to tell which (if any) attributes are common to the multiple data sources. A further issue for heterogeneous cases is that attributes that refer to the same real-world values may have different column headings between data sources (i.e. "*First Name*", "*1st Name*", "*Name-1*", "*Birth Name*" or "*Given Name*"). Therefore, multiple data sources may erroneously be declared as sharing no common attributes when they in fact do but under different headings. In order to allow for improved *Token-Blocking* for heterogeneous cases, one would need to identify the common attributes without referencing the schema. Papadakis et al. [51] achieve this by combining *Token-Blocking* with attribute clustering. Attribute clustering is a form of schema matching in which a similarity value is assigned to attribute column pairs between datasets. Highly similar attribute columns are clustered together and thought of as referring to the same value. *Token-Blocking* is then performed in a

way that only records that share a common token by clustered attributes are grouped together. In the experimental evaluations of [51], attribute clustering combined with *Token-Blocking* was seen to obtain near-equal PC to that of just *Token-Blocking* but in substantially less comparisons in almost every case.

4.2.3 Sorted Neighbourhood

Sorted Neighbourhood [8, 22, 56, 74] is a simple sort and merge style unsupervised blocking method. Records are first sorted in lexicographical order according to their respective values from a user-defined sorting key (i.e. a concatenation of substring values from an attribute or number of attributes). A window of fixed size is then passed over the ordered sorting key values. Only records that fit within the window at any given time are considered in the linkage phase. As no single sorting key is likely to perfectly order the matches of a dataset adjacently, multiple sorting keys tend to be used in independent passes as part of a *Multi-Pass Sorted Neighbourhood* approach [4, 48]. Figure 4.6 demonstrates *Sorted Neighbourhood* applied to the same records of Fig. 4.2 using windows of size 3. The sorting key value in this case consists of the first letter of the *First Name* attribute value combined with the last two characters of $Postcode$ and the entire of Age (highlighted in green).

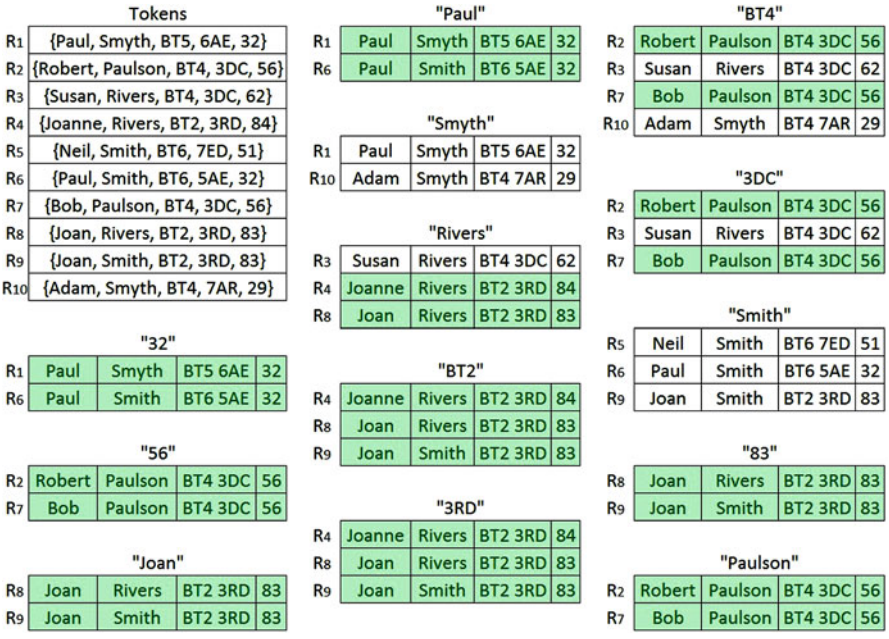


Fig. 4.5 Example of *Token-Blocking*. For clarity, only those blocks that contain more than one record, and thus form at least one record pair, are presented

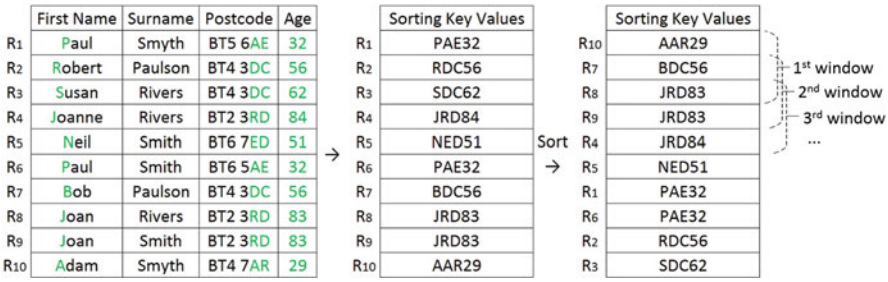


Fig. 4.6 Example of sorted neighbour

Sorted Neighbourhood is an easy to comprehend and implement blocking approach that dates back over 20 years, making it a staple go-to blocking method in the world of *RL*. In [26], *Sorted Neighbourhood* was seen to generate fewer false negatives than standard blocking and outperformed all other baseline approaches in terms of computational runtime on average. However, in the same paper *Sorted Neighbourhood* was also seen to generate more unnecessary comparison pairs than some of the other approaches. Another limitation of *Sorted Neighbourhood* is its dependency upon parameter choices such as window size and the sorting key(s). A small window size ensures a low number of comparisons but increases the risk of overlooking matching record pairs that do not happen to be positioned near one another in an ordered list. Conversely, a large window size increases the number of matching record pairs formed, but also the number of non-matching record pairs. Attributes with low diversity are poor contributors to a sorting key as many records could have the same attribute value (i.e. gender). If the number of records with the same sorting key value is substantially larger than the window size, many matching record pairs could be overlooked. In cases of sensitive data, a schema may not be available making the manual definition of a sorting key difficult. An additional downside of the *Sorted Neighbourhood* approach is the computational demand in ordering lists of sorting key values. For very large datasets, the ordering may become a computational bottleneck.

Variants of the *Sorted Neighbourhood* approach have been developed that attempt to further improve upon its efficiency and effectiveness by overcoming some of these disadvantages. One popular variant uses dynamic window sizing [14, 33, 41, 56, 61, 68, 69, 73, 75] to overcome the issues associated with using windows of a fixed size. Sorting key values are ordered lexicographically, as per the original approach, and each value is compared to those preceding it in the ordered list until a predetermined distance threshold is exceeded. Sorting key values that fall within the distance threshold are considered "close", as are their respective records. Records associated with "close" sorting key values collectively form record pairs for linkage. Dynamically sized windows are, in practice, often much smaller than a fixed window size, but increase in size when necessary.

In [58, 61], the dynamic window sizing variant of *Sorted Neighbourhood* is used to form index trees by which RL queries can be made in sub-second time. This type

of RL process is referred to as Real-Time RL and is discussed in greater detail in Sect. 4.2.7.

In [36], the authors note that *Sorted Neighbourhood* assumes that a sorting key can be applied to all records of a dataset (or datasets) as they all share the same attributes (i.e. schema). This is not always the case, such as with Resource Description Framework (RDF) data. Therefore, the authors adapt *Sorted Neighbourhood* to make it applicable. They achieve this by defining sorting keys based on the properties in the input RDF graphs, the subject Uniform Resource Identifiers (URI) in the respective triples or both of them. This is discussed in greater detail in Sect. 4.2.8.

In [57], a variation of *Sorted Neighbourhood* is proposed in which the ordered sorting key values of a dataset have a window of size 2 passed over them. Window sizes are dynamically and iteratively increased if any duplicates are detected within them. By iteratively extending window size in this manner, windows only ever become as large as necessary (i.e. until no further matches are found).

In [29], a variation of *Sorted Neighbourhood* referred to as *Sorted Neighbourhood on Encrypted Fields (SNEF)* is introduced which has the advantage of being applicable to encrypted fields. Rather than being a blocking method in its own right, it is an implementation that improves the quality of the record pairs that are generated by an independent blocking method (a concept referred to as *Meta-Blocking* which is explained in greater detail in Sect. 4.2.6). SNEF assigns each blocked record a score according to a rank function which is based on the results of the used blocking process. Records within each block are ranked accordingly, and a window of fixed size is passed over them. The intuition being that records within blocks that have similar rank scores are indicated as being especially similar. Records that fit within this window form record pairs for comparison as per the general *Sorted Neighbourhood* approach.

To better scale *Sorted Neighbourhood* for large datasets, it has been combined with the *MapReduce* framework [39, 40, 44, 46]. During the *Mapping* stage of the MapReduce framework, the workload of finding sorting key values for different records is partitioned so that it can be ran in parallel by many computers, rather than sequentially by a single computer. During the subsequent *Reduce* stage, the records are partitioned to each computer by sorting key values and then reduced into pairs that share the same sorting key value.

Collectively, these variations overcome many of the disadvantages of *Sorted Neighbourhood*, but parameter selection and the need for a domain expert tend to remain an issue. In many cases, there is still a reliance upon a domain expert to define the, potentially multiple, sorting keys that are needed. In [5], the automatic learning of sorting keys was explored but assumes the existence of labelled data. Additionally, although the dynamic window sizing approaches avoid selection of a fixed window size, there is still a need to select a distance threshold value by which the windows change size dynamically. This entails many of the same issues associated with selecting a fixed window size, albeit to a much lesser extent.

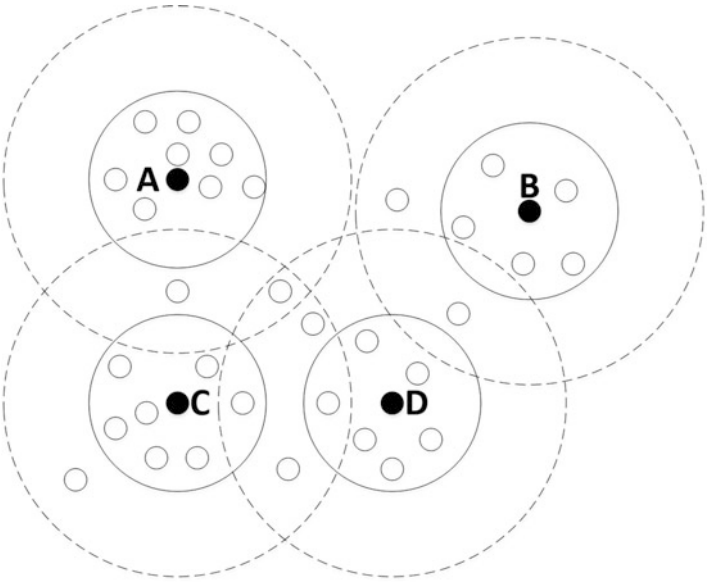


Fig. 4.7 Example of canopy clustering

4.2.4 Clustering-Based Blocking Methods

Clustering is a process by which records are divided into groups so that each record is similar to the records within the same cluster (similar according to a given similarity function) and dissimilar to records from different clusters. Overlapping clusters may be constructed if records are permitted to be assigned to more than one cluster. Adapted versions of the general clustering algorithm have been used as an alternative to blocking or to aid part of the blocking process.

Canopy Clustering [3, 6, 32, 35, 45] is an adapted version of the general clustering algorithm, that forms overlapping clusters of similar records. In *Canopy Clustering*, a record is chosen (potentially at random) as a centroid, and all other records within a *loose* distance threshold of it form a canopy. The centroid of a canopy and all records within a $tight$ distance threshold of it form an inner circle of especially similar records, that are excluded from becoming the centroid of any new canopy. Additional (non-excluded) records are chosen as centroids, and new canopies are formed in the same manner. This is best illustrated in Fig. 4.7 by centroids A, B, C and D having an inner circle of $tight$ distance threshold records and outer circle of $loose$ distance threshold records (i.e. $threshold_{loose} > threshold_{tight}$). Note that the $tight$ records of the different canopies never overlap. This ensures welcome overlapping of diverse canopies that cover many records collectively.

A disadvantage of *Canopy Clustering* is that its efficiency and effectiveness are very much reliant upon appropriate selection of parameters, namely the

loose and *tight* distance threshold values and the number of centroids. In [4], *Canopy Clustering* is shown to outperform standard blocking and sorted neighbourhood when appropriate parameters are selected; however, selection of such parameters can be difficult. Overly, loose distance threshold values may cause very large clusters to be generated resulting in poor RR and exceptionally long linkage runtime. On the other hand, selecting overly strict distance threshold values can result in very small clusters to be generated, which may fail to capture many of the matches within a dataset resulting in poor PC. *Canopy Clustering* complexity is dependent upon the selected number of centroids, as every other record must be compared to each of the centroids during the *Canopy Clustering* process. A smaller number of centroids therefore increase the efficiency of the clustering process, but at a potential cost to RR as large clusters may be generated. Conversely, a larger number of centroids may improve PC, especially if overlapping clusters are permitted, but at a cost to clustering efficiency. Therefore, poor parameter selection may result in the formation of a small number of large clusters (poor RR) containing most matching pairs (good PC), or a large number of small clusters (good RR) containing few matching pairs (poor PC). Determining which records to use as centroids is equally important. If multiple records referring to the same real-world entity are used as centroids of different clusters, then many matching record pairs may be overlooked, especially if clusters are not allowed to overlap.

In [18], a clustering approach based on the string similarity between BKVs is used to iteratively form blocks with regulated size. Between each iteration, a block quality and block size trade-off is used to determine which overly small blocks should be merged in order to improve *PC* and which overly large blocks should be further partitioned in order to improve *RR*. The authors acknowledge that although their BSL approach takes longer than that of the baselines, they achieve superior RR because of the regulated block sizes. They also detail that in their evaluations they manually determined the blocking keys, and the order in which they should further partition the overly large blocks. Automatic learning of blocking keys was left for future work.

4.2.5 Locality-Sensitive Hashing

Locality-Sensitive Hashing (LSH)-based blocking approaches [11, 19, 25, 31, 38, 42, 50, 64, 67, 71] use several functions selected from a family of hash functions to convert records into hash key values of a fixed size and placed into buckets of similar records according to their respective hash key values. By stipulating that the family of hash functions are locality-sensitive to a distance metric d, one ensures that the probability of generating the same hash key value by a hash function is dependent upon the distance between both records. Therefore, similar records are likely to generate the same hash key value by a particular hash function. In this section, we describe LSH when used with Jaccard Similarity, but other variants of LSH may be adapted for use with other distance metrics.

Definition 6 (Jaccard Similarity) The Jaccard Similarity of two sets of elements, A and B, is defined as the ratio of the size of their intersection to the size of their union (i.e. $\frac{A \cap B}{A \cup B}$)

Records are first divided into sets of substrings of length k referred to as k-shingles, a similar process to that of *q-Gram Indexing* discussed in Sect. 4.2.2. Just like with *q-Gram Indexing*, a desirable k value is one small enough so that typographical errors between near-similar record pairs are accounted for, but also large enough so that the generated shingle sets are of reasonable size so complexity does not become an issue.

For large datasets, storing all shingle sets of the respective records would be inefficient as there would be many repeated elements of length k. Instead, the shingle sets are used to form a sparse Boolean matrix in which the rows represent the universal shingle set (with each element thought of as a hash function, $h()$) and the columns represent each record. 1s and 0s are used to represent if the respective record of a column does or does not contain the respective shingle element of each row. Therefore, highly similar columns contain 1's for many of the same hash functions; which also means that they, and their respective records, have high Jaccard Similarity. In Fig. 4.8, we see a Boolean matrix constructed for the same records as that of Fig. 4.2 when using shingles of size $k = 2$ (i.e. Bi-Grams). For clarity, only a small sample of the universal 2-shingle set for this case is presented.

	First Name	Surname	Postcode	Age	2-shingles
R1	Paul	Smyth	BT5 6AE	32	{Pa, au, ul, Sm, my, yt, th, BT, T5, 5_, _6, 6A, AE, 32}
R2	Robert	Paulson	BT4 3DC	56	{Ro, ob, be, er, rt, Pa, au, ul, ls, so, on, BT, T4, 4_, _3, 3D, DC, 56}
R3	Susan	Rivers	BT4 3DC	62	{Su, us, sa, an, Ri, iv, ve, er, rs, BT, T4, 4_, _3, 3D, DC, 62}
R4	Joanne	Rivers	BT2 3RD	84	{Jo, oa, an, nn, ne, Ri, iv, ve, er, rs, BT, T2, 2_, _3, 3R, RD, 84}
R5	Neil	Smith	BT6 7ED	51	{Ne, ei, il, Sm, mi, it, th, BT, T6, 6_, _7, 7E, ED, 51}
R6	Paul	Smith	BT6 5AE	32	{Pa, au, ul, Sm, mi, it, th, BT, T6, 6_, _5, 5A, AE, 32}
R7	Bob	Paulson	BT4 3DC	56	{Bo, ob, Pa, au, ul, ls, so, on, BT, T4, 4_, _3, 3D, DC, 56}
R8	Joan	Rivers	BT2 3RD	83	{Jo, oa, an, Ri, iv, ve, er, rs, BT, T2, 2_, _3, 3R, RD, 83}
R9	Joan	Smith	BT2 3RD	83	{Jo, oa, an, Sm, mi, it, th, BT, T2, 2_, _3, 3R, RD, 83}
R10	Adam	Smyth	BT4 7AR	29	{Ad, da, am, Sm, my, yt, th, BT, T4, 4_, _7, 7A, AR, 29}

	Records									
2-shingles	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10
32	1	0	0	0	0	1	0	0	0	0
56	0	1	0	0	0	0	1	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
3D	0	1	1	0	0	0	1	0	0	0
an	0	0	1	1	0	0	0	1	1	0
Sm	1	0	0	0	1	1	0	0	1	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
th	1	0	0	0	1	1	0	0	1	1
yt	1	0	0	0	0	0	0	0	0	1

Fig. 4.8 Example of a Boolean matrix

Different hash functions from a Boolean matrix can be combined to form hash table functions by which records can be placed into hash tables. A single hash table function cannot be expected to perfectly partition a dataset into hash tables. Therefore, multiple hash table functions are often required in order to achieve good blocking results.

LSH is attractive as a blocking approach for a number of reasons. It can have time complexity as low as $O(n)$, making it highly scalable for even large and highly dimensional datasets. LSH may also be leveraged over similarity spaces (e.g. semantic similarity, and textual similarity) to improve block quality by removing repeated or redundant pairs without negatively affecting accuracy [71]. As individual records are converted into Boolean vectors the approach lends itself to privacy-preserving record linkage. With only alphanumeric characters (i.e. $A \rightarrow Z$ and $0 \rightarrow 9$) and spaces, the maximum number of unique k-shingles is limited to $(26 + 10 + 1)^k$. However, in reality the actual number is often much smaller as many k-shingles are unlikely to occur in real-world datasets.

Manual parameter setting is often needed when using LSH (e.g. k for k-shingling, and L for the number of hash table functions). The authors of [38] note that despite LSH's fast and effective performance it suffers from a number of drawbacks that make it unsuitable for large-scale record linkage. In particular, they highlight the memory requirement during the hash table generation process and the high runtime for datasets with many duplicates. Other papers have modified or adapted LSH as part of more complex blocking approaches that aim to overcome some of these drawbacks. A simple Hamming distance-based LSH variation is described in [19, 31, 42]. In this variation, values in randomly and uniformly chosen ith positions of the universal shingle set are used to construct hash table functions. In the same papers, the authors describe a Euclidean distance-based variation of LSH, in which vectors of records are projected onto a Euclidean space allowing for nearby points to be grouped together. This concept of mapping entries to a multidimensional vector space has been well documented in older works such as FastMap [17], StringMap [27], MetricMap [70] and SparseMap [24].

In the proceeding subsection, we give special attention to a particular variation of LSH that has been adopted in a considerable number of recent research efforts.

4.2.5.1 LSH with MinHashing

$MinHashing$ allows for the formation of a much smaller representation of a Boolean matrix, that still allows for a comparable estimation of the Jaccard similarity of two records by their respective columns. The $MinHash$ function, $h(C)$, takes the value of the first row in which 1 appears in a column C. Applied to the Boolean matrix of Table 4.1, we get $\{1, 3, 1, 1\}$ for columns $\{C_1, C_2, C_3, C_4\}$, respectively (note 1 appears in the top row of columns C_1, C_3 and C_4 but in the third row for column C_2). If the rows of a Boolean matrix are randomly permuted ($Orderings$ in Table 4.1), the probability that a row's hash function assigns two columns the same $MinHash$ value is approximately the same as their Jaccard

Table 4.1 Example of a Boolean matrix converted into a signature matrix

	Boolean matrix				Orderings								
Shingles	C_1	C_2	C_3	C_4	O_1	O_2	O_3						
s_1	1	0	1	1	2	2	4						
s_2	1	0	1	1	3	10	6			Signature matrix			
s_3	0	1	0	0	6	4	3			C_1	C_2	C_3	C_4
s_4	1	0	1	1	5	3	7		O_1	1	4	1	1
s_5	0	1	0	0	4	7	5	→	O_2	2	1	1	2
s_6	1	0	1	1	1	5	1		O_3	1	3	1	1
s_7	1	0	1	1	7	8	2						
s_8	0	1	1	0	8	1	9						
s_9	0	1	0	0	9	9	10						
s_{10}	0	1	0	0	10	6	8						

Similarity. $MinHash$ signatures are generated for each record by concatenating the individual $MinHash$ values for each record across multiple random row order permutations. In other words, the number of permutations dictates the length of the $MinHash$ signature value.

The Jaccard Similarity of a pair of $MinHash$ signatures (fraction of $MinHash$ values in which they agree) is approximately equal to that of the Jaccard similarity of their respective columns. Similar $MinHash$ signatures are therefore indicative of similar record pairs. The higher the number of permutations used to create the $MinHash$ signatures, the lower the expected error. In Table 4.1, C_1 and C_4 have identical $MinHash$ signatures, and looking at the Boolean matrix we see that this is also the case for their Boolean vectors. Blocking only the pairs with exactly matching signatures would be naive as even near-similar record pairs may have different $MinHash$ values for some hash functions (e.g. $\{C_1, C_3\}$ and $\{C_3, C_4\}$). Instead, the columns of the signature matrix are separated into l bands of m $MinHash$ values each. Columns containing the same $MinHash$ vector in each band are then blocked together as they match for that part of the signature.

MinHash LSH has an advantage over standard LSH in that different hash table functions no longer require definition. Instead, the $MinHash$ function is used iteratively upon a Boolean matrix with its rows permuted between each iteration. l and m values require definition for banding the signature matrix. However, these are arguably easier parameters to set with than the large number of hash table functions that potentially exist as part of the basic LSH approach. This is especially true for cases in which the universal shingle set is of considerable size.

MinHash LSH has been used in a considerable number of recent research papers [11, 31, 64, 71] due to it being fast, effective and relatively easy to implement. In [64], the authors note that due to the semi-structured nature of data of the Web, some data source schemas can potentially consist of thousands of attributes. They acknowledge that the comparison of all attribute pairs for the purpose of scheme matching is impracticable and, therefore, propose a *MinHash LSH*-based

preprocess to reduce this comparison space significantly. The set of attributes is represented as a matrix where each column is a vector of the attributes determined by a weight function. By iteratively permuting the rows of the matrix and concatenating the different $MinHash$ values for each column, $MinHash$ signatures can be constructed for the attributes represented by each column. The $MinHash$ signatures of each attribute are then partitioned into l bands of size m. Only the attributes of columns that agree by at least one $MinHash$ signature band are then passed onto a subsequent attribute-match induction algorithm, thus reducing the comparison space significantly. In [11, 71], *MinHash LSH* is combined with semantic constraints in order to better pair records based on both their textual and conceptual similarity. In adding an additional filter by which record pairs must agree, the comparison space formed by *MinHash LSH* is further reduced with little to no negative impact upon PC. This improves RL quality overall. In [31], LSH was combined with a homomorphic matching technique in order to allow for privacy-preserving record linkage. The authors of this paper also state that in their experimental evaluation, a Hamming distance variation of the LSH approach was able to achieve superior results than that of the popular *MinHash LSH* variation. In [38], the authors extend *MinHash LSH* to form an iterative match-merge LSH approach that reuses a single hash table, greatly reducing the hash table generation time. As records are placed into their respective $MinHash$ buckets, they are checked against any already placed records. If the newly placed record contains some or all of an already placed record, a merge or replacement occurs accordingly. This process iteratively continues (for all records) until a termination condition is met; that is, the reduction ratio of the ith iteration is below a predetermined threshold. Every match-merge step that is performed results in both a smaller record set and hash table. This is reflected in the respective experimental evaluations in which this extended variation was observed to operate up to 5.6 times faster than standard LSH whilst maintaining equivalent accuracy.

4.2.6 Meta-blocking and Progressive-Blocking

Meta-Blocking methods [12, 30, 49, 51, 53, 55, 64] aim to improve the efficiency of how linkage is performed following a blocking process. This may be achieved by restructuring the blocks prior to linkage (*Meta-Blocking*), or managing how they are processed during linkage (*Progressive-Blocking*), rather than by simply performing all record pair comparisons arbitrarily.

Meta-Blocking improves the linkage efficiency of a block collection by prioritising the most promising blocks or record pairs. In [51], the authors observe that a record paired with many other records within a block collection is unlikely to match with any of them during linkage. To omit such record pairs from the linkage process would therefore be useful as the number of comparisons could be reduced with negligible cost to recall. *Meta-Blocking* approaches commonly accomplish this by building a graph to represent the entity-to-block relationships of a block collection.

In such a graph, nodes represent the individual records of a block collection, and edges between records indicate co-occurrence of these records in at least one block. Information from this graph can then be leveraged to indicate the most similar record pairs within the blocks. In [51, 53, 55, 64], *Meta-Blocking* has been used to further improve the RL efficiency of the unsupervised and schema-agnostic $token$ blocking approach by only performing linkage upon blocks or record pairs indicated as most promising. The linkage process then continues until a cost/gain approximation indicates finding further matches to be too costly. In [64], the authors combine attribute clustering, $token$ blocking and *Meta-Blocking* to form an unsupervised blocking approach they refer to as the "BLAST Approach". The blocking and *Meta-Blocking* stages are evaluated with Pairs Quality (PQ) (Eq. (4.7)) used as a surrogate for precision alongside PC and $F_{PQ,PC}$, where tp is the number of true positives (matches correctly classified as matches) and fp is the number of false positives (non-matches incorrectly classified as matches).

$$PQ = \frac{|tp|}{|tp| + |fp|}. \tag{4.7}$$

In their experimental evaluations, PQ was seen to greatly improve, whilst maintaining high PC. Their approach was observed to perform equal to or better than other blocking and *Meta-Blocking* approaches.

Progressive-Blocking techniques aim to learn and adapt as they progress so as to improve the likelihood of detecting additional duplicates with minimal additional effort. A progressive *Sorted Neighbourhood* approach [57] was described earlier in Sect. 4.2.3 in which small windows that contain duplicates are extended in size in the hope of detecting further duplicates. Iterative blocking [23, 52, 55, 72] is another progressive approach in which any record pair classified as a match during linkage is merged and distributed to all other blocks containing either record, replacing the original record in the process. This implementation likely improves PC, because what would have been previously overlooked matches now have a higher chance of being detected due to the transitive relation of matching record pairs. Efficiency also tends to improve in a progressive convergent manner as the repeated merging, distribution and replacement of individual records with merged records in earlier blocks saves the overall processing time of subsequent blocks.

4.2.7 Blocking for Real-Time RL

The approaches presented so far in this chapter describe unsupervised and semi-supervised blocking methods for batch RL of structured relational datasets. In batch RL, the general RL process from Fig. 4.1 is enacted in its entirety upon a dataset or datasets. Ideally, placing most matching record pairs in blocks with few non-matching pairs so that the computational demand of the subsequent linkage process is reduced. Real-Time RL, on the other hand, involves a query record

being matched against records from possibly multiple, typically disparate, large data sources. For batch RL, runtime can be expected to be considerable if the datasets are of substantial size or complexity, but in real-time RL queries are ideally processed in sub-second runtime. In this section, we detail some real-time RL blocking techniques that help achieve this goal.

In [43], an LSH blocking approach (Sect. 4.2.5) is made scalable to large-scale dynamic datasets and query-able by combining it with a dynamic sorting tree. In this approach, the records of large LSH blocks are sorted, and a window is passed over them in order to return the nearest neighbours of a query record. The authors note that as *MinHash LSH* involves a random permutation of rows when forming LSH blocks, a predefined fixed sorting key may result in an entire block being returned as a query result. To overcome this issue, the authors define a scoring function by which one or several attributes that contain many unique and well-distributed values may be selected as a sorting key. In their experimental evaluations, the authors demonstrate that combining LSH with dynamic sorting trees results in much lower query times than that of LSH alone. They further state that their approach can be effective and efficiently used for large-scale real-time RL cases. This is particularly true for noisy data, as LSH operates on a k-Gram level.

In [58], a forest-based sorted neighbourhood indexing technique is proposed that aims to facilitate real-time RL by using multiple distinct index trees with different sorting keys. In this approach, an initial $build$ phase constructs index trees using records from an existing database. A subsequent $query$ phase then allows for the built index to be queried and all relevant entries to be retrieved with the index updated in the process. The tree data structure consists of braided trees, referred to as AVL trees [62], that combine a height balanced binary tree property with a double-linked list. Every node in the tree is linked to its alphabetically sorted predecessor and successor nodes, with a list of identifiers of all records that have the respective nodes key value as their sorting key value. The nodes of the tree are sorted alphabetically by these sorting key values where a sorting key value is generated for each record in the database, and a record identifier is inserted into the tree based on the sorting key value. Record identifiers of records with matching sorting key values are listed together to the same node as a list. This allows for sorting key value searching to be reduced from $O(\log(n))$, as per array-based indexing, to $O(\log(k))$ given that there are k different nodes in a tree, n records in a dataset and $k < n$. Multiple different sorting key values are used during the build phase to construct multiple trees with records inserted into every tree where applicable as a node. A query record is inserted into all of the respective nodes of the different trees according to the sorting key used to generate them. For a single tree, all candidate records of all nodes within a window to the node containing the query record are considered candidate matches to the query record. This repeats for all trees forming a collective set of candidate matches to the query record. In their experimental evaluations, the authors experiment with both fixed and adaptive window size. The authors state that by using multiple trees blocking is noticeably improved over use of single trees with only a minor increase to the average insertion and query time. In the worst case, the achieved times are still considerably fast, that

is ∼1 ms insertion time and ∼15 ms query time, respectively, when using three trees. In their experimental evaluations, they found that their approach was able to perform over one order of magnitude faster than that of a similar baseline approach [60]. The same authors of [58] improve upon this work in another paper [59] by combining it with automatic learning of keys used for constructing the multiple trees. They achieve this by generating weakly labelled training data using the approach of [34] which was described earlier in Sect. 4.2.1. With their weakly labelled data, they evaluate individual keys and select those deemed most appropriate for real-time RL according to a scoring function that considers three factors: key coverage, generated block sizes and distribution of block sizes. The ideal keys are those that have high coverage, low average block size and low variance between block sizes. In their experimental evaluations, they evaluate the real-time RL approach of their previous paper using keys selected by this technique, against those selected by the blocking key selection technique of [34]. Note that in the baseline paper optimal keys were selected for standard blocking not real-time RL. They find that their selection technique chooses significantly better keys for real-time RL than the baseline in terms of query time, with comparable recall for two of the three evaluation datasets and only a 5% decrease over the baseline for the third dataset.

4.2.8 Blocking for RDF Data

Resource Description Framework (RDF) is a series of specifications for conceptually describing or modelling information among Web resources. Web information may be represented as an RDF triple. That is, a statement in the form of a *Subject–Predicate–Object* expression where a $Subject$ is connected to an $Object$ by a $Predicate$ (a verb) that details the type of relationship between the two. $Predicates$ may take multiple values unlike attributes in relational RL [2]. Representing Web information resources in this manner allows for their blocking and interlinking among multiple knowledge bases. Much recent work has focused on the blocking and linking of RDF data, a process commonly referred to as instance matching or link discovery.

In [32], a two-step blocking scheme learner for scalable linkage discovery is presented. With the proposed approach, an optional unsupervised dataset mapping is first performed between a pair of dataset collections (arbitrary mixes of RDF and tabular datasets). Matrices are generated using the dot product of the normalised term (individual token) frequency vectors of two datasets. Pairs between collections are then mapped according to a confidence function of the max Hungarian algorithm [28] applied to a respective matrix. Following this, Mapped pairs proceed to a second step which learns a link-discovery blocking scheme. RDF datasets are reconciled with the tabular datasets by representing them as property tables. From this point onwards, the problem is reduced to the more common problem of learning effective blocking schemes for tabular datasets with differing schemas. The authors use the same BSL approach as that of their earlier work in [34]

(discussed in Sect. 4.2.1) and incorporate bagging (random sampling) in case there are an insufficient number of training examples. In cases where there are no training examples at all, they suggest using the automatic training set generation algorithm presented in the same earlier paper. In their experimental evaluations, perfect mapping was achieved in all three test sets. Furthermore, their approach outperformed the baseline regardless of whether it was completely unsupervised or provided with a perfectly labelled training set. As this approach incorporates much of [34], it has many of the same advantages and disadvantages that were discussed in Sect. 4.2.1.

In [65, 66], an approach is presented that selects candidate pairs by discriminating keys chosen using domain-independent unsupervised learning. The same authors of both papers present this work for RDF, but state in [66] that it generalises to structured datasets too. Starting with a set of triples, all data-type $Predicates$ and instances are extracted. For each $Predicates$, all $Object$ values are retrieved, and three metrics ($Discriminability$, $Coverage$ and $F_{Dis,Cov}$) are computed. $Discriminability$ gives an indication of the diversity of a $Predicate$ and is defined as the ratio of its number of unique respective $Object$ values to its total number of respective $Object$ values. Any $Predicates$ with $Discriminability$ less than a pre-defined threshold value are immediately omitted from consideration as many instances have the same $Object$ values on this $Predicate$ which indicates poor RR. $Coverage$ is the ratio of the number of instances a key covers to that of the total number of instances, indicating the $Coverage$ of a $Predicate$. $F_{Dis,Cov}$ is the harmonic mean of $Discriminability$ and $Coverage$ and is used to indicate how well a $Predicate$ achieves a balance of being highly discriminative whilst covering a large number of instances. The $Predicate$ with the highest $F_{Dis,Cov}$ value is selected as the candidate selection key, provided that it is higher than a predetermined threshold value. If no suitable $Predicate$ is found, then conjunctions of $Predicates$ are iteratively explored until one is. Triples are indexed according to the selected key and when selecting candidate pairs only those with *n-Gram* similarity above a pre-defined threshold value are returned. In both papers, the proposed method outperforms the baselines but the authors admit to a number of potential problems. Firstly, they cannot be sure what effect values such as phone numbers may have on the size of the returned candidate sets. This is due to how their approach retrieves candidate pairs based on common *n-Gram*'s, and is never evaluated using numerical data. Given data primarily describing instances in the same geographic area, one can expect many phone numbers to share the same prefix; therefore, oversized candidate sets would be expected. Another issue they state is that as their algorithm looks for exact matching on query tokens when looking up similar instances within the index that not all co-referent instances for a given instance may be retried in poor quality datasets. They leave addressing this issue with fuzzy matching as an area for future research.

4.3 Conclusion

In this chapter, we discussed different categories of unsupervised and semi-supervised blocking methods giving examples of each and stating their advantages and disadvantages. A number of simple blocking approaches were initially presented that have been used for many years now. These simple approaches have been shown to perform especially well when a domain expert with intrinsic knowledge of the target data is at hand as they can effectively set the necessary parameters.

It was also discussed that under different circumstances (e.g. sensitive data, lack of domain expert, RDF data and RL queries) the simple blocking approaches may no longer be suitable therefore requiring more advanced approaches specific to these circumstances. For each circumstance, it was explained how different advanced blocking approaches came to be and are applicable where simple blocking approaches are not. The ultimate goal of unsupervised blocking research is for the development of unsupervised blocking approaches that are completely free of parameters. The approaches presented in this chapter may be unsupervised but operating completely parameter-free persists as a goal to be achieved. Areas of research that have arose as a result of blocking research were also discussed, namely *Meta-Blocking*, *Progressive Blocking* approaches and blocking approaches for RDF data. As datasets become increasingly larger and we look to the Web for their storage, one can predict that these particular areas of research will become increasingly relevant. With such exponential growth, the development of unsupervised blocking approaches for exceptionally large-scale data is expected to continue as a relevant topic of research.

References

1. Aizawa, A., Oyama, K.: A fast linkage detection scheme for multi-source information integration. In: Proceedings of International Workshop on Challenges in Web Information Retrieval and Integration, WIRI'05, pp. 30–39. IEEE, Piscataway (2005)
2. Atencia, M., David, J., Scharffe, F.: Keys and pseudo-keys detection for web datasets cleansing and interlinking. In: International Conference on Knowledge Engineering and Knowledge Management, pp. 144–153. Springer, Berlin (2012)
3. Babu, B.V., Santoshi, K.J.: Unsupervised detection of duplicates in user query results using blocking. In: International Journal of Computer Science and Information Technologies, **5**(3), 3514–3520. IJCSIT (2014)
4. Baxter, R., Christen, P., Churches, T., et al.: A comparison of fast blocking methods for record linkage. In: ACM SIGKDD, vol. 3, pp. 25–27. Citeseer (2003)
5. Bertolazzi, P., De Santis, L., Scannapieco, M.: Automatic record matching in cooperative information systems. In: Proceedings of the International Workshop on Data Quality in Cooperative Information Systems (DQCIS), p. 9 (2003)
6. Bilenko, M., Kamath, B., Mooney, R.J.: Adaptive blocking: learning to scale up record linkage. In: Sixth International Conference on Data Mining, ICDM'06, pp. 87–96. IEEE, Piscataway (2006)

7. Christen, P.: Improving data linkage and deduplication quality through nearest-neighbour based blocking. In: Department of Computer Science, The Australian National University. ACM (2007)
8. Christen, P.: Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer, Berlin (2012)
9. Christen, P.: A survey of indexing techniques for scalable record linkage and deduplication. IEEE Trans. Knowl. Data Eng. **24**(9), 1537–1555 (2012)
10. Christen, P., Churches, T., et al.: Febrl-freely extensible biomedical record linkage. In: Department of Computer Science, Australian National University (2002)
11. Cui, M.: Towards a scalable and robust entity resolution-approximate blocking with semantic constraints. In: COMP8740: Artificial Intelligence Project, Australian National University (2014)
12. dal Bianco, G., Gonçalves, M.A., Duarte, D.: Bloss: effective meta-blocking with almost no effort. Inf. Syst. **75**, 75–89 (2018)
13. Bilenko, M.: Learnable similarity functions and their application to clustering and record linkage. In: Proceedings of the Nineteenth National Conference on Artificial Intelligence, pp. 981–982 (2004)
14. Draisbach, U., Naumann, F., Szott, S., Wonneberg, O.: Adaptive windows for duplicate detection. In: 2012 IEEE 28th International Conference on Data Engineering (ICDE), pp. 1073–1083. IEEE, Piscataway (2012)
15. Duda, R.O., Hart, P.E., Stork, D.G., et al.: Pattern Classification, 2nd edn, p. 55. Wiley, New York (2001)
16. Elfeky, M.G., Verykios, V.S., Elmagarmid, A.K.: Tailor: a record linkage toolbox. In: Proceedings of 18th International Conference on Data Engineering, pp. 17–28. IEEE, Piscataway (2002)
17. Faloutsos, C., Lin, K.I.: FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets, vol. 24. ACM, New York (1995)
18. Fisher, J., Christen, P., Wang, Q., Rahm, E.: A clustering-based framework to control block sizes for entity resolution. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 279–288. ACM, New York (2015)
19. Gionis, A., Indyk, P., Motwani, R., et al.: Similarity search in high dimensions via hashing. In: VLDB'99 Proceedings of the 25th International Conference on Very Large Data Bases, vol. 99, pp. 518–529 (1999)
20. Gravano, L., Ipeirotis, P.G., Jagadish, H.V., Koudas, N., Muthukrishnan, S., Srivastava, D., et al.: Approximate string joins in a database (almost) for free. In: Proceeding VLDB '01 Proceedings of the 27th International Conference on Very Large Data Bases, vol. 1, pp. 491–500 (2001)
21. Guillet, F., Hamilton, H.J.: Quality Measures in Data Mining, vol. 43. Springer, Berlin (2007)
22. Hernández, M.A., Stolfo, S.J.: Real-world data is dirty: data cleansing and the merge/purge problem. Data Min. Knowl. Disc. **2**(1), 9–37 (1998)
23. Herschel, M., Naumann, F., Szott, S., Taubert, M.: Scalable iterative graph duplicate detection. IEEE Trans. Knowl. Data Eng. **24**(11), 2094–2108 (2012)
24. Hristescu, G., Farach-Colton, M.: Cluster-preserving embedding of proteins. Technical Report 99-50, Computer Science Department, Rutgers University (1999)
25. Ioannou, E., Papapetrou, O., Skoutas, D., Nejdl, W.: Efficient semantic-aware detection of near duplicate resources. In: Extended Semantic Web Conference, pp. 136–150. Springer, Berlin (2010)
26. Isele, R.: Learning expressive linkage rules for entity matching using genetic programming. Ph.D. thesis (2013)
27. Jin, L., Li, C., Mehrotra, S.: Efficient record linkage in large data sets. In: Proceedings of the Eighth International Conference on Database Systems for Advanced Applications, DASFAA '03, p. 137. IEEE Computer Society, Washington (2003). http://dl.acm.org/citation.cfm?id=789081.789250

28. Jonker, R., Volgenant, T.: Improving the Hungarian assignment algorithm. Oper. Res. Lett. **5**(4), 171–175 (1986)
29. Karakasidis, A., Verykios, V.S.: A sorted neighborhood approach to multidimensional privacy preserving blocking. In: IEEE 12th International Conference on Data Mining Workshops (ICDMW), pp. 937–944. IEEE, Piscataway (2012)
30. Karakasidis, A., Koloniari, G., Verykios, V.S.: Scalable blocking for privacy preserving record linkage. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 527–536. ACM, New York (2015)
31. Karapiperis, D., Verykios, V.S.: An LSH-based blocking approach with a homomorphic matching technique for privacy-preserving record linkage. IEEE Trans. Knowl. Data Eng. **27**(4), 909–921 (2015)
32. Kejriwal, M., Miranker, D.P.: A two-step blocking scheme learner for scalable link discovery. In: CEUR Workshop Proceedings. Vol 1317, pp. 49–60 (2014)
33. Kejriwal, M., Miranker, D.P.: N-way heterogeneous blocking. Tech. rep., TR-14-06 (2013)
34. Kejriwal, M., Miranker, D.P.: An unsupervised algorithm for learning blocking schemes. In: IEEE 13th International Conference on Data Mining (ICDM), pp. 340–349. IEEE, Piscataway (2013)
35. Kejriwal, M., Miranker, D.P.: On linking heterogeneous dataset collections. In: International Semantic Web Conference (Posters & Demos), pp. 217–220. Citeseer (2014)
36. Kejriwal, M., Miranker, D.P.: Sorted neighborhood for schema-free RDF data. In: International Semantic Web Conference. pp. 217–229. Springer, Berlin (2015)
37. Kejriwal, M., Miranker, D.P.: An unsupervised instance matcher for schema-free RDF data. Web Semant. Sci. Serv. Agents World Wide Web **35**, 102–123 (2015)
38. Kim, H.S., Lee, D.: Harra: fast iterative hashed record linkage for large-scale data collections. In: Proceedings of the 13th International Conference on Extending Database Technology, pp. 525–536. ACM, New York (2010)
39. Kolb, L., Thor, A., Rahm, E.: Parallel sorted neighborhood blocking with MapReduce (2010). arXiv:1010.3053
40. Kolb, L., Thor, A., Rahm, E.: Multi-pass sorted neighborhood blocking with MapReduce. Comput. Sci. Res. Dev. **27**(1), 45–63 (2012)
41. Lehti, P., Fankhauser, P.: Unsupervised duplicate detection using sample non-duplicates. Lect. Notes Comput. Sci. **4244**, 136 (2006)
42. Leskovec, J., Rajaraman, A., Ullman, J.D.: Mining of Massive Datasets. Cambridge University Press, Cambridge (2014)
43. Liang, H., Wang, Y., Christen, P., Gayler, R.: Noise-tolerant approximate blocking for dynamic real-time entity resolution. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 449–460. Springer, Berlin (2014)
44. Ma, K., Yang, B.: Parallel NoSQL entity resolution approach with MapReduce. In: International Conference on Intelligent Networking and Collaborative Systems (INCOS), pp. 384–389. IEEE, Piscataway (2015)
45. McCallum, A., Nigam, K., Ungar, L.H.: Efficient clustering of high-dimensional data sets with application to reference matching. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 169–178. ACM, New York (2000)
46. Mestre, D.G., Pires, C.E., Nascimento, D.C.: Adaptive sorted neighborhood blocking for entity matching with MapReduce. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing, pp. 981–987. ACM, New York (2015)
47. Michelson, M., Knoblock, C.A.: Learning blocking schemes for record linkage. In: AAAI'06 Proceedings of the 21st National Conference on Artificial Intelligence, pp. 440–445 (2006)
48. Naumann, F., Herschel, M.: An introduction to duplicate detection. Synth. Lect. Data Manage. **2**(1), 1–87 (2010)
49. Papadakis, G., Palpanas, T.: Blocking for large-scale entity resolution: challenges, algorithms, and practical examples. In: IEEE 32nd International Conference on Data Engineering (ICDE), pp. 1436–1439. IEEE, Piscataway (2016)

50. Papadakis, G., Ioannou, E., Niederée, C., Palpanas, T., Nejdl, W.: Eliminating the redundancy in blocking-based entity resolution methods. In: Proceedings of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries, pp. 85–94. ACM, New York (2011)
51. Papadakis, G., Ioannou, E., Palpanas, T., Niederee, C., Nejdl, W.: A blocking framework for entity resolution in highly heterogeneous information spaces. IEEE Trans. Knowl. Data Eng. **25**(12), 2665–2682 (2013)
52. Papadakis, G., Koutrika, G., Palpanas, T., Nejdl, W.: Meta-blocking: taking entity resolution to the next level. IEEE Trans. Knowl. Data Eng. **26**(8), 1946–1960 (2014)
53. Papadakis, G., Papastefanatos, G., Koutrika, G.: Supervised meta-blocking. Proc. VLDB Endowment **7**(14), 1929–1940 (2014)
54. Papadakis, G., Alexiou, G., Papastefanatos, G., Koutrika, G.: Schema-agnostic vs schema-based configurations for blocking methods on homogeneous data. Proc. VLDB Endowment **9**(4), 312–323 (2015)
55. Papadakis, G., Papastefanatos, G., Palpanas, T., Koubarakis, M.: Scaling entity resolution to large, heterogeneous data with enhanced meta-blocking. In: 19th International Conference on Extending Database Technology, pp. 221–232 (2016)
56. Papadakis, G., Svirsky, J., Gal, A., Palpanas, T.: Comparative analysis of approximate blocking techniques for entity resolution. Proc. VLDB Endowment **9**(9), 684–695 (2016)
57. Papenbrock, T., Heise, A., Naumann, F.: Progressive duplicate detection. IEEE Trans. Knowl. Data Eng. **27**(5), 1316–1329 (2015)
58. Ramadan, B., Christen, P.: Forest-based dynamic sorted neighborhood indexing for real-time entity resolution. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, pp. 1787–1790. ACM, New York (2014)
59. Ramadan, B., Christen, P.: Unsupervised blocking key selection for real-time entity resolution. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 574–585. Springer, Berlin (2015)
60. Ramadan, B., Christen, P., Liang, H., Gayler, R.W., Hawking, D.: Dynamic similarity-aware inverted indexing for real-time entity resolution. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 47–58. Springer, Berlin (2013)
61. Ramadan, B., Christen, P., Liang, H., Gayler, R.W.: Dynamic sorted neighborhood indexing for real-time entity resolution. J. Data Inf. Qual. **6**(4), 15 (2015)
62. Rice, S.V.: Braided AVL trees for efficient event sets and ranked sets in the SIMSCRIPT III simulation programming language. In: Western MultiConference on Computer Simulation, San Diego, pp. 150–155 (2007)
63. Shu, L., Chen, A., Xiong, M., Meng, W.: Efficient spectral neighborhood blocking for entity resolution. In: IEEE 27th International Conference on Data Engineering (ICDE), pp. 1067–1078. IEEE, Piscataway (2011)
64. Simonini, G., Bergamaschi, S., Jagadish, H.: Blast: a loosely schema-aware meta-blocking approach for entity resolution. Proc. VLDB Endowment **9**(12), 1173–1184 (2016)
65. Song, D., Heflin, J.: Scaling data linkage generation with domain-independent candidate selection. In: Proceedings of the 10th International Semantic Web Conference (ISWC) (2013)
66. Song, D., Heflin, J.: Automatically generating data linkages using a domain-independent candidate selection approach. In: International Semantic Web Conference, pp. 649–664. Springer, Berlin (2011)
67. Steorts, R.C., Ventura, S.L., Sadinle, M., Fienberg, S.E.: A comparison of blocking methods for record linkage. In: International Conference on Privacy in Statistical Databases, pp. 253–268. Springer, Berlin (2014)
68. Tamilselvi, J.J., Gifta, C.B.: Handling duplicate data in data warehouse for data mining. Int. J. Comput. Appl. **15**(4), 1–9 (2011)
69. Tamilselvi, J., Saravanan, V.: Token-based method of blocking records for large data warehouse. Adv. Inf. Mining **2**(2), 5–10 (2010)

70. Wang, J.T.L., Wang, X., Lin, K.I., Shasha, D., Shapiro, B.A., Zhang, K.: Evaluating a class of distance-mapping algorithms for data mining and clustering. In: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 307–311. ACM, New York (1999)
71. Wang, Q., Cui, M., Liang, H.: Semantic-aware blocking for entity resolution. IEEE Trans. Knowl. Data Eng. **28**(1), 166–180 (2016)
72. Whang, S.E., Menestrina, D., Koutrika, G., Theobald, M., Garcia-Molina, H.: Entity resolution with iterative blocking. In: Proceedings of the 2009 ACM SIGMOD International Conference on Management of data, pp. 219–232. ACM, New York (2009)
73. Whang, S.E., Marmaros, D., Garcia-Molina, H.: Pay-as-you-go entity resolution. IEEE Trans. Knowl. Data Eng. **25**(5), 1111–1124 (2013)
74. Winkler, W.E.: Overview of record linkage and current research directions. Bureau of the Census. Citeseer (2006)
75. Yan, S., Lee, D., Kan, M.Y., Giles, L.C.: Adaptive sorted neighborhood methods for efficient record linkage. In: Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 185–194. ACM, New York (2007)

Chapter 5
Traffic Sensing and Assessing in Digital Transportation Systems

Hana Rabbouch, Foued Saâdaoui, and Rafaa Mraihi

Abstract By integrating relevant vision technologies, based on multiview data and parsimonious models, into the transportation system's infrastructure and in vehicles themselves, the main transportation problems can be alleviated and road safety improved along with an increase in economic productivity. This new cooperative environment integrates networking, electronic, and computing technologies, will enable safer roads, and achieve more efficient mobility and minimize the environmental impact. It is within this context of digital transportation systems that this chapter attempts to review the main concepts of intelligent road traffic management. We begin by summarizing the most best-known vehicle recording and counting devices, the major interrelated transportation problems, especially the congestion and pollution. The main physical variables governing the urban traffic and factors responsible for transportation problems as well as the common assessing methodologies are overviewed. Graphics and real-life shots are occasionally used to clearly depict the reported concepts. Then, in direct relation to the recent literature on surveillance based on computer vision and image processing, the most efficient counting techniques published over the few last years are reviewed and commented. Their few drawbacks are underlined and the prospects for improvement are briefly expressed. This chapter could be used not only as a pedagogical guide, but also as a practical reference which explains efficient implementing of traffic management systems into new smart cities.

H. Rabbouch (✉)
Université de Tunis, Institut Supérieur de Gestion de Tunis, Cité Bouchoucha, Tunis, Tunisia

F. Saâdaoui
University of Monastir, Laboratoire d'Algèbre, Théorie de Nombres et Analyse Non-linéaire, Faculté des Sciences, Monastir, Tunisia

R. Mraihi
Université de Manouba, Ecole Supérieure de Commerce de Tunis, Campus Universitaire de La Manouba, Tunis, Tunisia

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_5

5.1 Introduction

Since antiquity, transportation has been one of the most vital factors contributing to the global economic growth. In the beginning, water and land lanes were the dominant transportation paradigms. With industrial revolutions came improvements in road construction, the railways, and the airplane [10]. Today, in the digital information age, the challenge is for global management efficiency in a new world governed by economic alliances, environmental concerns, and logistic and trade agreements. Thus, in this digital-focused world, effective transportation management has never been as important as it is right now. In the new transportation systems, several interacting and related objectives are to be considered and need to be optimized, such as efficient traffic management, efficient moving of people, and freight, reduce transportation-related emissions, and accurate prediction of real-time vehicle and pedestrian flows. Intelligent transportation systems (ITS) have emerged as the cost-effective technology that have the potential to support all these objectives [57]. The ITS are drastically changing the way we commute by reducing traffic congestion and vehicle collisions, both of which significantly improve road safety, while decreasing pollution. The ITS encompass a multitude of techniques from fundamental management systems such as traffic signal control systems, containers management, plate recognition, or speed cameras to monitor applications, such as closed-circuit television systems. With the rise of Information Technology and Communications (ITC), leading to high resolution digital shooting techniques and accessibility of intelligent beacon sensors, it becomes possible to enhance the technical capabilities to facilitate benefits from transportation systems throughout the world. We especially note the emergence of smart applications that include online data processing and feedback from other transportation sources, such as smart vehicles and parkings, geographical information systems (GIS), and weather information [15].

By carefully integrating relevant technologies, based on both reliable data sets and significant and parsimonious models, into the transportation system's infrastructure and in vehicles themselves, the main above-mentioned transportation problems can be alleviated and road safety improved along with an increase in productivity. This new dynamic interface integrates networking, electronic, and computing technologies, and will enable safer roads, and achieve more efficient mobility and minimize the environmental impact [15]. On the other hand, fusing heterogeneous traffic information is becoming a major support leading to a fastest growth of the ITS [20]. A variety of data collecting and communicating methods used in ITS, such as sensors and communication networks, bring about huge volumes of data that need to be implemented in the main transportation applications and the related innovation on smart cities. Indeed, in order to provide more accurate decisions about the traffic on a road network, the conventional traffic sensors used to measure the prevailing traffic need to well interconnected and better optimized. Several sources of data, such as cameras, radars, GPS, and cell phone tracking, could be used to complement the information provided by conventional

techniques. Multiple sources provide complementary data, and multi-source data fusion can produce a better understanding of the observed situation by decreasing the uncertainty related to the individual sources. On this basis, developing a data-driven ITS could offer a well-adapted answer to the operational needs of traffic management centers and traffic information operators, allowing them to achieve their goal more efficiently.

It is within this context of digital transportation systems that this chapter attempts to review the main concepts of the road traffic management and ITS. We begin by summarizing the most best-known vehicle recording and counting devices, the major interrelated transportation problems, especially the congestion and pollution. The main physical variables governing the urban traffic and factors responsible for transportation problems as well as the common assessing methodologies are overviewed. Graphics and real-life shots are occasionally used to clearly depict the reported concepts. Then, in direct relation to the recent literature on surveillance based on computer vision and image processing, the most efficient counting techniques published over the few last years are reviewed and commented. Their few drawbacks are underlined and the prospects for improvement are briefly expressed. This chapter could be used not only as a pedagogical guide, but also as a practical reference which explains efficient implementing of traffic management systems into new smart cities. The structure of the chapter is as follows. A summary of the literature about essential problems related to transportation, especially those of congestion and/or pollution phenomena, and some decision-support solutions, is put forth in Sect. 5.2. Then, a background on the physical key factors commonly used for assessing urban traffic flows is defined in Sect. 5.3. Section 5.4 provides descriptions of the main traffic measurement devices, while Sects. 5.5 and 5.6 focus on the vision-based sensors and their related statistical/mathematical models. The final section is dedicated to the main perspectives and concluding remarks.

5.2 Main Transportation Problems and Decision-Support Solutions

In the context of economic globalization, the need for movement of goods and people has experienced significant growth. This increasing mobility is one of the main causes of traffic congestion. Nowadays, reducing congestion is considered as one of the major challenges in transportation, especially in large cities. Technically, traffic congestion is the result of too many vehicles crowding the road space available with the lack of alternative travel options. It takes time and energy, causes pollution and stress, decreases productivity, and imposes costs on society. The traffic congestion involves queuing, slower speeds, and travel times increased, which impose costs on the economy and generate multiple impacts on urban areas and their inhabitants. Congestion also has a range of indirect impacts, including quality of life, security, and impacts on road space for non-vehicular users, such as users

of sidewalks and facade properties of the road [47]. It is a major problem that most cities face and therefore, many measures have been taken to limit congestion. It is believed that the identification of congestion characteristics is the first step in this effort, because it is an essential guidance in choosing appropriate measures. Given the cycle of transportation demand, it is not always possible to solve congestion by increasing capacities [50]. In this context, many authorities are responsible for the development of efficient strategies to manage demand as well in time as space, and also discourage the growth if needed. Despite this, urban congestion has become, over the few last years, more costly in terms of time, money, and fuel.

From a literary perspective, the most important researches have appeared just recently. Bharadwaj et al. [8] showed that traffic congestion on city roads not only increases the fuel consumption but consequently leads to increase in carbon dioxide emissions, outdoor air pollution as well as increase in the exposure time of the passengers. According to [12], for the top hundred largest US urban centers, congestion generated 4.8 billion hours of travel delays in 2011, up from 1.1 billion hours in 1982. Congestion also required 8.419 million cubic meters of excess fuel consumption in 2011, up from 1.73 million cubic meters in 1982. Finally, the excess CO2 emitted from congestion amounted to 19.524 billion kilograms in 2011, up from 3.94 billion kilograms in 1982. Lu et al. [36] examined the congestion and pollution consequences of driving-to-school trips. Their suggestion to transport policymakers was to lower such congestion and environmental costs via optimizing the spatial balance between school supply and demand. Shi et al. [52] used the detrended cross-correlation analysis to investigate relationships between NO2 pollution and traffic congestion. Wu et al. [61] found a significant impact of congestion charging on traffic and emissions in Beijing. Furthermore, recent researches have revealed that even economies are negatively affected by congestion. Hymel [26] showed that high levels of congestion dampen employment growth. Jin and Rafferty [29] reexamined the relationship and proved that congestion also negatively affects income growth. Using macroscopic traffic simulations and vehicle emissions calculation, Wu et al. [61] assessed the impact of congestion charging on traffic and emissions in Beijing.

To face these problems, data-driven decision-making models have considerably grown with the progress of the ICT. Measuring and analyzing tools have significantly risen, hence providing immediate and reliable information about traffic flows, resulting in the emergence of the concept of ITS. Such intelligent devices are capable of providing decision makers with real-time measurements of the traffic at a given point of the road network. Once appropriately fused and analyzed, these heterogeneous data can be exploited to address many concerns for: (1) mobility, with instant diffusion of traffic conditions, achieving automatic toll or the intelligent management of lights and priorities traffic, planning of new infrastructure; (2) security, with automatic incident detection or monitoring of vehicle tunnels, and (3) environmental and public health, with pollution control and infrastructure regulation (lighting, tunnel ventilation). In the literature, several techniques have been proposed for the purpose of multi-sensor fusion under heterogeneous data configurations. Due to the different types of sensors that are used and the heterogeneous

nature of information that needs to be combined, different data fusion techniques are being developed to suit the applications and data. These techniques were drawn from a wide range of areas including artificial intelligence, pattern recognition, statistical estimation, and other areas.

Treiber et al. [55] developed an advanced interpolation method for estimating smooth spatio-temporal profiles for local highway traffic variables such as flow, speed, and density. The method was based on an adaptive smoothing method which takes as input stationary detector data as typically collected by traffic control centers. Their method was also generalized to allow for fusion with floating car data or other traffic information. Faouzi et al. [20] provided a survey of how data fusion is used in different areas of ITS. Anand et al. [2] used a data fusion approach based on the Kalman filtering which brings the advantages of both spatial and location-based data for the estimation of traffic density. Subsequently, the estimated data were utilized for predicting density forward to future time intervals using a time series regression model. In Shan et al. [51], an incomplete traffic data fusing method was proposed to estimate traffic state. Their approach improves missing data estimation by extracting data correlations and applying incomplete data fusion, implementing the two approaches in parallel. The main research focus is on extracting the inherent spatio-temporal correlations of traffic states data from road segments based on a multiple linear regression model. He et al. [25] made an investigation into the fusion of a new data combination from cellular handoff probe system and microwave sensors. And a fusion method based on the neural network technique was proposed. To identify the factors influencing the accuracy of fusion results, they analyzed the sensitivity of those factors by changing the inputs of neural network-based fusion model. Their experiments shown that handoff link length and sample size were identified as the most influential parameters to the precision of fusion. Weibin et al. [60] proposed a data fusion method to merge traffic speed from different data sources according to their prior probability that can be inferred from a high-order multivariable Markov model, which is itself developed in a systemic perspective.

5.3 Road Traffic Properties

5.3.1 Traffic Components

In general, the road traffic [17] consists of two main components: (1) infrastructure and (2) moving objects. The infrastructure is a set of interconnected structural elements which form the framework to support all the traffic structure. For example, for roads and motorway networks, infrastructure contains structures such as bridges, culverts, signage and markings, electrical systems (street lights and traffic), and edge treatments (curbs, sidewalks, landscaping). The moving objects are individuals who use infrastructure, mainly said vehicles. Pedestrians are not considered because their movements are supposed to depend on moving objects. Note that stopped or parked

vehicles, as defined in road rules, are considered as a part of the infrastructure, and generally, are not taken into account. In other words, only objects in motion on the road and interact with signals are considered. In the following subsections, we introduce the basic variables and the usual measures for traffic analysis. Vehicles can be roughly categorized into four modes: the two and three wheels vehicles, heavy goods vehicles (HGV), public transports, and light vehicles (LV). In this chapter, we will consider, as in Cohen [16] and [17], that vehicles are divisible, i.e., they can be expressed passenger car units (PCU), which corresponds in French to Unité de Véhicule Particulier.

5.3.2 Elementary Variables of the Traffic

The considerable development of the traffic and the constant increase in car ownership have necessitated the development of the theory of traffic engineering to be able to describe, explain, and predict the interactions of vehicles as well as overall movements in road infrastructure. This theory, be it deterministic or probabilistic, is introduced through a number of variables, relationships, or distribution of characteristics [16]. Below are the main elementary variables of the traffic.

5.3.2.1 Inter-vehicular Time Gap

At a given point of the road, the inter-vehicular time gap [17, 48] is the time that elapses between the moment of passage of two successive vehicles (see Fig. 5.1). This variable is particularly useful in the studies related to road safety and in

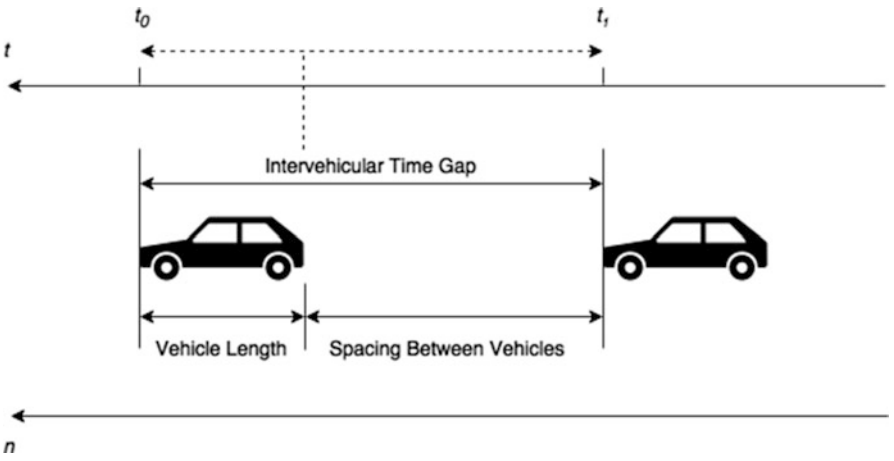


Fig. 5.1 Inter-vehicular time gap

many traffic simulation problems [6]. Nowadays, several real-time algorithms for regulating crossroads with lights in city use the inter-vehicular time gap. This parameter allows to understand the effects of the composition of the traffic on the flow traffic conditions. Experimental observation enables obtaining empirical distributions of the gaps. Hence, in addition to the well-known features of position and dispersion, the empirical distribution of the gaps provides several useful indicators, some of which are the proportion of short-intervals (threshold level below which the traffic is considered dangerous) and the equivalence coefficient e (PCU), which expresses each class of vehicles as a number of passenger cars [17]. The characteristics of the gap distributions vary according to the type of the road, traffic level, composition, weather conditions and visibility, etc.

5.3.2.2 Flow Rate

The flow rate corresponds to the distribution of vehicles in time. As defined in [16] and [17], the average flow q at the point x between t_1 and t_2 is given as,

$$q(t_1, t_2, x) = \frac{\sharp(t_1, t_2, x)}{t_2 - t_1}, \tag{5.1}$$

where $\sharp(t1, t2, x)$ denotes the number of vehicles past by x between the two instants. Experimentally, the flow rate can be determined by simple counting (traditionally by humans) on the road. In mathematics and physics, the flow of vehicles is commonly considered continuous. We then define the flow function $q(x, t)$ at point x and at time t as,

$$q(x, t) = \lim_{\delta t \to 0} q\left(\frac{2t - \delta t}{2}, \frac{2t + \delta t}{2}, x\right). \tag{5.2}$$

This definition does not apply to a discrete traffic flow analysis, since the limit would be either infinite or zero, depending on a vehicle passes at time t or not. Obviously, there is a convergence of $q(t - \frac{\delta t}{2}, t + \frac{\delta t}{2}, x)$ towards $q(x, t)$ for small values of δt. To make the connection with the microscopic approach, let us mention that the average flow rate is equal to the average of the inverse of the inter-vehicular time gap (here denoted h) for a stationary flow (flow rate which does not vary much around its average). If we consider N the number of inter-vehicular times observed for a period T at a given point of the road, we have,

$$T = \sum_{i=1}^{N} h_i = N \times \bar{h}, \tag{5.3}$$

where $\bar{h}$ denotes the average time interval. Therefore, the estimated flow rate q_i is

$$q = \frac{N}{T} = \frac{1}{\bar{h}}. \tag{5.4}$$

As commented in [17], it is commonly admitted in the literature that the maximum flow rate can reach about 0.5 vehicles per second.

5.3.2.3 Traffic Density

The traffic density can be defined as the number of vehicles per unit length of the roadway. The average density $k(x_1, x_2, t)$ at time t over a limited section of the road between two points x_1 and x_2 corresponds to the ratio,

$$k(x_1, x_2, t) = \frac{\sharp(x_1, x_2, t)}{x_2 - x_1}, \tag{5.5}$$

where $\sharp(t_1, t_2, x)$ denotes the number of vehicles in the section at time t. This physical quantity can be measured using aerial photography or video cameras [17]. The physical mathematics theory defines the continuous concentration $k(x, t)$ at the point x and at time t as,

$$k(x, t) = \lim_{\delta t \to 0} k\left(\frac{2x - \delta x}{2}, \frac{2x + \delta x}{2}, t\right). \tag{5.6}$$

5.3.2.4 Occupancy Rate

The occupancy rate is a dimensionless quantity, defined as the proportion of time during which the loop remains occupied. The occupancy rate, often denoted τ, is directly related to the density k, the mean length of vehicles L, and the length of the sensor l, with $\tau = k(L + l)$. Nowadays, several researches in the field of logistics, urbanism, and tourism use the occupancy rate as fundamental variable to explain many vital factors of the modern transport policy [3, 31, 44]. The magnetic loops are commonly used to collect occupancy rate data. The magnetic loops are well-known sensors often embedded within the roadway, and sensitive to changes of the magnetic field produced by the passage metal masses of the vehicles.

5.3.2.5 Vehicle Speed

The speed is the distance covered per unit time [59]. For each vehicle, recording the instantaneous speed can characterize the temporal profile of the speed. It is one of the main indices characterizing the road traffic and is frequently used for assessing

the energy consumption of vehicles. Over a course of period T, the average speed of a vehicle is defined as,

$$\bar{v} = \frac{1}{T}\int_0^T v(t)dt, \tag{5.7}$$

where $v(t)$ is the instantaneous speed of the vehicle at time t. However, it is not convenient to track the speed for each vehicle. Thus, average speeds are measured by sampling vehicles in a given area over a period of time. Two main definitions of mean speed are identified: time mean speed (TMS) and space mean speed (SMS). At a fixed point of the road, the TMS can be considered as the arithmetic mean of instantaneous speeds v_i of N vehicles, passing during an indeterminate period of time,

$$v_t = \frac{1}{N}\sum_{i=1}^{N} v_i, \tag{5.8}$$

where $v_i = \frac{d_i}{t}$, with d_i the distance traveled by the ith vehicle during the period t. The SMS notion is more useful in practice and it is considered more accurate than the TMS. It is measured over the whole roadway segment. Consecutive images or video sequences of a roadway track the speed of individual vehicles, and then the average is calculated. The data for calculating the SMS may be taken from satellite images, traffic cameras, or both.

$$v_s = \left(\frac{1}{N}\sum_{j=1}^{N}(1/v_i)\right)^{-1} \tag{5.9}$$

where $v_j = \frac{d}{t_j}$, t_j is the time required for the vehicle j to travel the distance d. N represents the number of vehicles passing the roadway segment. Thus, the SMS can be viewed as the harmonic mean of the speeds.

5.3.2.6 Fundamental Diagram of Traffic Flow

The fundamental diagram of traffic flow [16] represents the macroscopic relationship involving traffic flow, traffic density, and velocity. It is often called macroscopic fundamental diagram (MFD). The fundamental diagram is sensitive to many factors such as road geometry, nature and composition of traffic, weather conditions, and operating measures. It is commonly used to predict the capability of a road system, or its behavior when applying inflow regulation or speed limits. The traffic is assumed homogeneous and stationary. In other words, the flow rate, density, and speed vary slightly around their respective means, $\bar{q}$, $\bar{k}$, and $\bar{v}$. As depicted in Fig. 5.2, a low density corresponds to high speed of the flow. This speed is

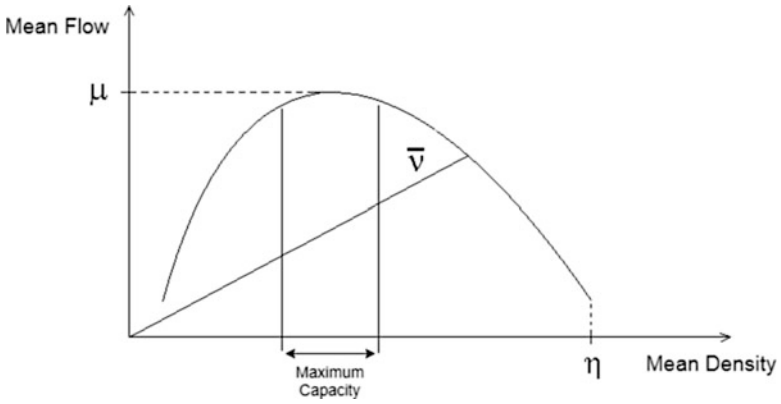


Fig. 5.2 Fundamental diagram of traffic flow

commonly called free speed. Inversely, as k increases, the dispersion between vehicles become more important, and consequently, v decreases. This suggests that v is a decreasing function of k. The fundamental diagram describes the capacity of the network in terms of vehicle density with μ being the maximum capacity of the network and η being the jam density of the network. The maximum capacity of the network is the region at the peak of the function.

5.3.3 Assessing Traffic Congestion

5.3.3.1 Main Attempts

Researches in this context have rapidly grown on the basis of the fundamental diagram. The aim was to give a more precise meaning to the concept of congestion by linking contributions of engineering to those of the economy, hence offering an effective identification and control of congestion. The desire to clarify the concept of congestion throughout externalities has led Kolm [32] to develop the principle of service quality, where he showed that the main cause is the accumulation of road users, leading finally to a lower transportation service quality. The quality of service is considered in terms of daytime according to the fundamental diagram. This notion of quality particularly interests us and is close to the general definition of Arnott and Kraus [4]. It allows to distinguish a request amount of a demand for quality and raises the question of arbitration between the two. The answer in the case of traffic congestion passes through the pricing. Kolm [32] formalizes congestion by the speed-flow function while providing nuance in determining the speed. He considers the speed according to the spacing between vehicles. Thus, congestion occurs when the speed of a vehicle is limited by that of the preceding one. In other words, the congestion function is based on the relationship between speed and vehicle spacing. It is composed of three parameters which together takes the form of a production

Fig. 5.3 Rains causing massive traffic jams in Riyadh

function of service: The first v expresses the speed, the second n_t measures the quantity of vehicles per unit of time served, while the third z reflects the offer in service considering v as a function of n_t. Therefore, congestion is expressed in time lost by the individuals based on the traffic density. The time factor is considered as determinant of the quality of service produced by an infrastructure.

Vickrey [56] gave a more accurate definition to the congestion which contradicts the notions of transport externality. The question is no longer to evaluate congestion but to analyze the flow conditions of roads including the most difficult places, such as nodes, strictures of ways, pavement jams, or weather as shown in Fig. 5.3. The congestion effect is seen as the difference between actual arrival time and the arrival time in unperturbed condition for movement between two landmarks X and Y. This approach seems effective, especially for problems where urban congestion depends heavily on time and individuals' behavior (anticipation, relocating, etc.) (see Laroche [34]). Finally, let us mention that Hau [24], attempting to characterize the capacity of a roadway, has shown that it varies little from one country to another, as long as only the tread is considered. It gives an average of 1000 vehicles per hour and per direction for a two-lane road and about 1800–2000 vehicles per hour and per direction for a four-lane road [34].

5.3.3.2 Congestion Measures

As discussed in [34, 49], most congestion indices are calculated from measurement of travel time (or delay). Both references distinguish between measure and index

in the sense that the measure is linked to a unit (additional travel time in minutes) while an index reads without. Most of the measurements are made by comparing the actual situation to the initial situation. The definitions point in the same direction as the National Cooperative Highway Research Program (NCHRP) Reports 398 [35] and 618 [11] that present the main congestion measures and indices as follows:

- Travel Rate (TR in minutes per mile): The TR is a direct indicator of the amount of travel time, which makes it relevant to travelers.

$$\text{TR (min/mile)} = \frac{\text{travel time (min)}}{\text{segment length (miles)}} \tag{5.10}$$

- Delay Rate (DR in minutes per mile): The DR is the rate of time loss for vehicles operating in congested conditions for a specified roadway segment or trip.

$$\text{DR (min/mile)} = \text{Actual Travel Rate} - \text{Acceptable Travel Rate} \tag{5.11}$$

- Relative Delay Rate (RDR): The RDR is a dimensionless measure that can be used as a congestion index to compare the relative congestion on facilities, modes, or systems in relation to different mobility standards for system elements such as freeways, arterial streets, and transit routes.

$$\text{RDR} = \frac{\text{Delay Rate}}{\text{Acceptable Travel Rate}} \tag{5.12}$$

- Delay Ratio (DR_a): The DR_a is a dimensionless measure that can be used to compare or combine the relative congestion levels on facilities with different operating characteristics like freeways, arterial streets, and transit routes.

$$DR_a = \frac{\text{Delay Rate}}{\text{Actual Travel Rate}} \tag{5.13}$$

- Corridor Mobility Index (CMI): The CMI consists of the speed of person movement value divided by some standard value, such as one freeway lane operating at nearly peak efficiency with a typical urban vehicle occupancy rate. This may be one method of addressing the magnitude and relativity problems with the speed of person movement.

$$\text{CMI} = \frac{\text{Passenger Volume} \times \text{Average Travel Speed}}{\text{Optimum Facility Value}} \tag{5.14}$$

- Travel Rate Index (TRI): This index allows to compare measured travel rates to free flow conditions for any combination of streets and freeways. The values can be related to the public as an indicator of the length of extra time spent in the transportation system during a trip.

$$\mathrm{TRI} = \frac{\left[\frac{\mathrm{FTR}}{\mathrm{FFFR}} \times \mathrm{FPP}_{\mathrm{VMT}}\right] + \left[\frac{\mathrm{PASTR}}{\mathrm{PASFFR}} \times \mathrm{PASPP}_{\mathrm{VMT}}\right]}{[\mathrm{FPP}_{\mathrm{VMT}} + \mathrm{PASPP}_{\mathrm{VMT}}]}, \tag{5.15}$$

where acronyms are respectively FTR: Freeway Travel Rate; FFFR: Freeway Free Flow Rate; $\mathrm{FPP}_{\mathrm{VMT}}$: Freeway Peak Period Vehicle Miles Traveled; PASTR: Principal Arterial Street Travel Rate; PASFFR: Principal Arterial Street Free Flow Rate; and $\mathrm{PASPP}_{\mathrm{VMT}}$: Principal Arterial Street Peak Period Vehicle Miles Traveled.

- Buffer Time Index (BTI): The BTI is a measure of trip reliability that expresses the amount of extra buffer time needed to be on time for 95% of the trips. As with the TRI, indexing the measure provides a time- and distance-neutral measure, but the actual minute values could be used by an individual traveler for a particular trip length or specific origin–destination (O-D) pair.

$$\mathrm{BTI} = \frac{\left[\text{95th Percentile Travel Time} - \text{Average Travel Time}\right]\text{(min)}}{\text{Average Travel Time (min)}} 100\%, \tag{5.16}$$

In the NCHRP reports, it is mentioned that it is complex to develop a congestion measure that takes into account all the aspects that characterize congestion. Consequently, several measures have been developed. In most of the interesting methods, travel time measurement and delay are considered key elements in defining the congestion. The travel time is used to calculate the speed of travel and the delay on a route. The annual average daily traffic (AADT) is also an important information to quantify the demand on each section of the road network. From this information, it is relatively easy to define one or more indices with one or with a combination of these parameters (travel time, delay, speed, AADT or their derivatives: veh/km, etc.).

Moreover, among indexes which have been proposed, the only one which measures the full range of system performance and which allows comparisons across metropolitan areas is the roadway congestion index (RCI) developed by Texas Transportation Institute [54]. The RCI is a measure of vehicle travel density on major roadways in an urban area and it is composed as follows:

$$\mathrm{RCI} = \frac{\left[\begin{matrix}\text{Freeway}\\ \text{VMT(ln/mile)}\end{matrix} \times \begin{matrix}\text{Freeway}\\ \text{VMT}\end{matrix}\right] + \left[\begin{matrix}\text{PAS}\\ \text{VMT(ln/mile)}\end{matrix} \times \begin{matrix}\text{PAS}\\ \text{VMT}\end{matrix}\right]}{\left[a \times \begin{matrix}\text{Freeway}\\ \text{VMT}\end{matrix}\right] + \left[b \times \begin{matrix}\text{PAS}\\ \text{VMT}\end{matrix}\right]}, \tag{5.17}$$

where acronyms stand for: PAS VMT: Principal Arterial Street Vehicle Miles Traveled; PAS VMT (ln/mile): Principal Arterial Street Vehicle Miles Traveled per lane mile; Freeway VMT: Freeway Daily Vehicle Miles Traveled;

Freeway VMT (ln/mile): Freeway Daily Vehicle Miles Traveled per lane mile. *a* and *b* are two parameters varying depending on the type of lane. An RCI exceeding 1 indicates an undesirable congestion level, on an average, on the freeways and principal arterial street systems during the peak period.

5.4 Different Measurement Devices

There are generally two types of traffic sensors: intrusive systems (installed in the shod) and non-intrusive systems (installed on roadsides, or embedded within vehicles). Intrusive sensors are mostly inductive loops placed within each circulation channel, which react to the passage of vehicles [39]. The introduction of a double loop also allows measuring the speed and length of vehicles. The major drawback of inductive loops is the need to intervene in the road, and thus, disrupt the traffic for installation and maintenance. Moreover, the loops are limited to functions of counting and measuring and individual classification by traffic lane.

On the other hand, non-intrusive sensors are remote sensors using various technologies, such as radar, laser, ultrasound, and video. The radar is able to count and measure the speed of vehicles (by Doppler or by frequency modulation) on a strip of circulation whatever the climate. The radars are generally fairly expensive equipment. Lasers emit an energy beam which scans detecting surface (a driving lane). The transit time is measured and is used to model a profile of the vehicle. By using two successive detection planes, a laser system is also capable of measuring the vehicle speed. With the capture of the 3D profile of the vehicle, lasers allow a detailed and accurate classification of vehicles. They usually work for a single traffic lane and are quite expensive. Ultrasonic sensors work similarly to the radar but in a range of lower frequency. They can be disrupted by emissions of certain vehicles and have difficulty measuring fast movements. However, economic considerations make these sensors an ideal choice for operators.

Finally, sensors based on the vision analysis use a traffic camera to capture images and videos from road networks to be analyzed. A camera placed at a optimal distance can record up to two or three traffic lanes. A processing unit is then established in order to perform the interpretation of these sub-shots and extract the vehicles and assess their behavior. However, cameras operating in the traffic are commonly affected by adverse conditions that influence the visibility (night, rain, snow, fog, shadows, etc.). On the other hand, the camera is now a reliable and very inexpensive sensor, and secondly, it provides a representation (pictures) directly usable by humans because the signal is close to that perceived by the eye. It has been widely installed on roads these recent years for passive surveillance needs.

5.4.1 Pneumatic Sensors

Pneumatic sensors (see Fig. 5.6) can perform traffic counts, and thus measure flow rates. They consist of a rubber cable, stretched across the floor and connected to a detector. The crushing of the cable at the passage of a vehicle causes a pressure detected by a manometer activating a relay. It is then possible to count the number of axles passing over the sensor by cumulating pulses in a counter. The counters are then expressed in PCU. These sensors, still prevalent for road counts, have some advantages, including ease of installation, good portability of the sensor–detector assembly, and possibility of battery operation providing an autonomy of several days. However, the system is of a high average cost. In addition, the cable can be pulled off during the passage of heavy vehicles. In saturated state, imprecision can sometimes exceed 20%.

5.4.2 Electromagnetic Loops

The electromagnetic loop is considered among the most practical devices for measuring traffic parameters in many countries. It is polyvalent in its application since it can be installed both in towns and on highways and expressways. The sensor consists of an inductive loop embedded within the road surface. The passage of the metal mass of a vehicle over the loop causes a variation of the electromagnetic field. This variation results in a voltage pulse whose length is linked to the vehicle and its passage time. A single loop by way allows to measure flow rate observations but also the occupancy rate τ, defined by

$$\tau = \frac{100}{T} \sum_{i=1}^{N} \tau_i \tag{5.18}$$

where τ_i is the occupation time of the loop corresponding to the measure period i, and T is the total measurement time.

5.4.3 Acoustic Sensors: Ultrasonics

The acoustic sensor consists of a directional antenna fixed on a support. This antenna emits an ultrasound wave propagating with a known speed. When passing a vehicle, ultrasound wave strikes a reflective surface. A fraction of this wave reflected by the mobile, and is then sensed by the receiver after a certain detection time. This duration of detection time allows calculation of the occupancy rate. The detector also provides a counting of vehicles. The sensor may often be mounted on a gantry above and in the center of the traffic lane. Detection time is thus

variable according to vehicles height. This feature allows to discriminate several categories. A proper mounting avoids the reception of false echoes. In conditions of normal operation, the accuracy of distance measurement is ±0.5 m. The propagation velocity of ultrasound waves being dependent on the temperature and humidity of the air (Figs. 5.4, 5.5, 5.6, and 5.7).

5.4.4 Microwave Sensors

A microwave sensor [38] is a low-cost advanced sensor used for detecting and measuring traffic at intersections and on roadways. The term microwave refers to the wavelength of the transmitted energy, usually between 1 and 30 cm, corresponding to a frequency range of 1–30 GHz. This detector provides per-lane presence indication, as well as volume, occupancy, speeds, and classification information, in up to eight lanes or detection zones simultaneously. A microwave radar provides the option of multiple lane operation, but cannot detect stopped vehicles. Two types of

Fig. 5.4 Doppler microwave (TC26-B, Operating Frequency: 10.525 GHz) vehicle motion sensor (Source: MS Sedco company, Indianapolis, USA)

Fig. 5.5 Passive infrared radar traffic detector (Source: FA BEMA Mobile Traffic Lights)

Fig. 5.6 Pneumatic road tube traffic data recorder installed at Prince Saud Bin Mohammed Bin Muqrin Road in the north of Riyadh (Saudi Arabian capital)

Fig. 5.7 Permanent traffic camera at King Fahd Road, one of the main north-south roads of Jeddah (major urban center of western Saudi Arabia)

microwave sensors are used in traffic management applications, continuous wave (CW) Doppler radar and frequency modulated continuous wave (FMCW) radar. Doppler sensors are most commonly used. They transmit a signal that is constant in frequency with respect to time. According to the Doppler principle, the motion of a vehicle in the detection zone causes a shift in the frequency of the reflected signal. A photograph of Doppler microwave radar used for vehicle detection purposes can be seen in Fig. 5.4.

5.4.5 Infrared Sensors

Infrared (IR) sensors are electronic instruments that are used to capture certain characteristics of its neighborhood by either detecting and/or emitting infrared radiations. Two IR sensors are used for traffic flow modeling applications: (1) infrared sensors called active, illuminate detection zones with low-power infrared energy transmitted by laser diodes operating in the near infrared region of the electromagnetic spectrum at 0.85 mm. A part of the transmitted energy is reflected or scattered by vehicles back towards the sensor, and (2) passive infrared sensors transmit no energy of their own. Rather they detect: energy emitted from vehicles, road surfaces, and other objects in their field-of-view, and energy emitted by the atmosphere and reflected by vehicles, road surfaces, or other objects into the sensor aperture.

The energy captured by IR sensors is focused by an optical system onto an infrared-sensitive material mounted at the focal plane of the optics. This material transforms the reflected and emitted energy into electrical signals. Signals are transmitted to a processing stage, where they are analyzed for the presence of a vehicle. The sensors are mounted overhead to view approaching or departing traffic. They can also be mounted in a side-looking configuration [30]. IR sensors are utilized for signal control; volume, speed, and class measurement; detection of pedestrians in crosswalks; and transmission of traffic information to motorists. A picture of a passive infrared radar traffic detector is given in Fig. 5.5.

5.4.6 Video Sensors

The above-presented techniques are often electromagnetic and electronic detection systems that can sense a set of vehicles passing or arriving at a certain point. Such equipments are either buried under the roadway surface or embedded into the roadway. Thus, they are too cumbersome and cannot therefore be maintained. In comparison with old sensors, traffic cameras are installed on the roadside and are considered as a major component of most ITS. Monitoring centers receive live records and perform video analyses. Today, since cameras are easily operated, controlled, and maintained, the traffic video data have replaced old-fashioned

ones and are now extensively applied to resolve many other transport problems. It deserves to be mentioned that a higher accuracy could be expected through the integration of multiple data sources including both traditional and modern technologies. However, it is still a challenge to better integrate heterogeneous data and fusing them into a singular data scheme.

The video cameras have been introduced to advanced traffic management to ensure surveillance in the main arteries given their ability to transmit closed-circuit television imagery to a human operator for interpretation [1, 5, 7, 13, 22, 37]. These days, traffic managers use video and image processing to automatically analyze scenes of interest, and then, extract valuable information for traffic management and surveillance purposes (see Fig. 5.7). A video image processor system (VIPS) typically consists of one or several cameras, a microprocessor-based computer for digitizing and analyzing the imagery, and softwares for interpreting the images and converting them into traffic flow data. A VIPS can replace several in-ground inductive loops, provide detection of vehicles across several lanes, and perhaps lower maintenance costs. Some VIPS process data from more than one camera and further expand the area over which data are collected [30].

VIPS detects vehicles through the analysis of binary or color images or video sequences gathered by cameras at roadway sections. Binary imaging analysis is performed by image processing algorithms that examine the variation of gray levels in groups of pixels (picture elements) contained in the video frames. Several studies have been conducted on algorithms that are sensitive to color features, for example, those that act in eliminating shadow artifacts or enhance vehicle discrimination in adverse weather conditions (see Rabbouch et al. [45]). Along with vehicle class and size data, color fingerprints or signatures have been proposed to determine traffic volume, turning movements, lane changes, and link travel time by reidentifying a vehicle or group at a downstream site [30]. As discussed in [33, 41, 58], the main traffic video processing algorithms are designed to omit gray level or color variations in the stationary image background. These algorithms are supposed to also ignore variations caused by weather conditions, shadows, and daytime or nighttime artifacts, but retain objects identified as automobiles, trucks or buses, motorcycles, and bicycles. Traffic flow parameters are calculated by analyzing successive video frames.

5.5 Traffic Video Processing

5.5.1 Recent Research

Detecting moving objects in a sequence of images taken at different intervals is one of the important areas of computer vision. Several applications in different disciplines work on the detection of any moving object, such as video surveillance, remote sensing, medical treatment, and underwater detection. [23]. The area that

interests us in this chapter is video surveillance, in particular, the analysis of the images of a traffic scene. Generally, vehicles present an important tool in people's daily lives. It is for this reason that researchers are interested in studying road traffic. In this context, several methods have been proposed for detecting, tracking, and counting vehicles. We present a brief literature review of these methods in what follows.

The literature associated with the traffic video surveillance focuses on automatic vehicle counting (AVC) using basic image/video processing techniques, with the aim of obtaining useful statistics on roads exploitation. The paper [40] is dedicated to the detection and counting of vehicles in the day environment using real-time flow traffic through differential techniques. The basic idea used is the variation in traffic flow density due to the presence of a vehicle in the scene. In this work, a differential algorithm is designed to detect and count vehicles. With this method, the authors managed to control in real time the flow of traffic in urban areas. In [18], the authors present a method for detecting vehicles and a counting system based on digital image processing techniques. These images can be taken by IP cameras installed at the top of existing traffic lights. Using the proposed approach, it is possible to detect the number of vehicles waiting on each side of the intersection, in order to provide the necessary information for optimal traffic management. After integrating the proposed algorithms into a traffic management system, it was possible to reduce CO2 and fuel emissions by half compared to the standard fixed time scheduler.

More recently, Jang et al. [27] proposed an algorithm to detect and count vehicles passing by a certain point in the video of traffic flow monitoring. The particularity of this algorithm is to calculate an approximate value of the velocity while counting vehicles using mixture models for background modeling. Raghtate et al. [46] introduced a technique to avoid human surveillance and automate the video surveillance system. This technique avoids the need to have a background image of the traffic. For an input video signal given, the frames are extracted. The selected images are used to estimate the background. This background image is subtracted from each input video image and the foreground object is obtained. After post-treatment technique, the counting of vehicles is done. Xia et al. [62] proposed an EM-estimated Gaussian mixture model to improve quality segmentation of moving vehicles. In addition, a method of restoration is designed to eliminate noise. A morphological feature and color histogram are finally used to solve occlusion problems. Efficacy and efficiency experiments show that the proposed approach can improve the result of vehicle counting and vehicle occlusion detection. Finally, the research in [42] proposes a method based on a scale-invariant feature transform (SIFT) algorithm, to carry out the classification and counting of the vehicles. This algorithm allows the detection of remarkable points that will identify an object. These points are invariant to scaling, rotation, and translation as well as affine transformations and illumination changes. This improves the efficiency of classification and counting of vehicles. It is noticeable that other recent researches in this context are well discussed in [45].

5.5.2 Challenges in Modeling Traffic Scenes

For any indoor or outdoor scene, changes occur over time. Thus, it is primordial for any substantive model to be able to tolerate these changes. Nevertheless, these changes can be local, affecting only parts of the background, or global affecting the entire background. The analysis of these changes is important to understand the motivations behind different background subtraction procedures [9]. The possible changes in a background scene can be roughly classified according to their source. We essentially have:

- Illumination changes: which can be (1) gradual change in lighting, such as changing the relative location of the sun during the day, (2) sudden lighting changes, such as turning lights on or off in an indoor scene, and/or (3) shadows thrown on the background by objects in the background, for example, by moving objects in the foreground (moving shadows).
- Motion change: which are principally caused by (1) small movements of the camera, which cause an overall movement of the image. Indeed, despite the assumption that the cameras are stationary, the movements of small cameras due to the wind load, and (2) parts of the background are moving. For example, the branches of trees move with the wind.
- Structural change: These are changes introduced within the background, including any changes in the appearance or geometry of the scene background caused by the targets. For example, if someone moves something from the background, or if a car is parked in the scene or comes out of the scene.

5.6 Statistical Modeling of Traffic Video Data

In this section, we introduce some widely used statistical modeling approaches for traffic video data. For each model, we present initial conditions and main properties. For simplicity, the intensity of pixels is considered as observation.

5.6.1 Probabilistic Principle

The technique of background/foreground segmentation can be explained as follows. For a pixel, the observed intensity is a random variable that has a value based on whether the pixel belongs to the background or the foreground. Given the intensity observed at a pixel at time t, denoted x_t, we have to classify this pixel on the background $\mathscr{BG}$ or the foreground classes $\mathscr{FG}$. It is a subject of classification into two groups. In a Bayesian framework, we test whether this pixel belongs to

the background or to the foreground, based on the ratio between the posterior probabilities $p(\mathscr{BG}|x_t)/p(\mathscr{FG}|x_t)$. Using Bayes rule, we can compute posterior probabilities as

$$p(\mathscr{BG}|x_t) = \frac{p(x_t|\mathscr{BG})p(\mathscr{BG})}{p(x_t)}, \qquad p(\mathscr{FG}|x_t) = \frac{p(x_t|\mathscr{FG})p(\mathscr{FG})}{p(x_t)}, \tag{5.19}$$

where $p(x_t|\mathscr{BG})$ and $p(x_t|\mathscr{FG})$ are the likelihoods of observations x_t given the background and foreground models, respectively. The terms $p(\mathscr{BG})$ and $p(\mathscr{FG})$ are the prior beliefs that the pixel belongs to the background or the foreground [9]. If equal priors are assumed, the ratio between the posterior is reduced to the ratio between the probability, which is typically indicated by the likelihood ratio

$$\frac{p(x_t|\mathscr{BG})}{p(x_t|\mathscr{FG})}. \tag{5.20}$$

Hence, modeling the statistical background aims to provide estimates for the likelihood that the observation gives a model for the background and a model for the foreground. However, since the intensity of a foreground pixel can arbitrarily take any value, the foreground distribution is assumed to be uniform. Finally, the problem leads to a classification problem in one class, and a decision can be made by comparing the probability of observing the background model with a threshold,

$$\begin{cases} x_t \text{ belongs to } \mathscr{BG} & \text{if } p(x_t|\mathscr{BG}) \geq \varepsilon \\ x_t \text{ belongs to } \mathscr{FG} & \text{otherwise,} \end{cases} \tag{5.21}$$

where ε is a threshold parameter. Therefore, any background subtraction approach requires a statistical model to estimate the probability of observing data in the base class. In the following, we discuss parametric methods that assume a Gaussian model or a mixture of Gaussian model for the pixel process.

5.6.2 Gaussian Background Model

In background modeling, pixel intensity is the most commonly used characteristic in traffic video processing. In a static scene, a simple noise model that can be considered for the pixel process is the independent stationary additive Gaussian noise model [14]. Accordingly, the distribution of the noise at a defined pixel is a Gaussian distributed with a zero mean and σ standard deviation density, i.e., $\mathscr{N}(0, \sigma^2)$. Formally, the intensity observed at this pixel is a random variable with a Gaussian distribution

$$P(x_t|\mathscr{BG}) \approx \mathscr{N}(\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp \frac{(x_t - \mu)^2}{2\sigma^2}. \tag{5.22}$$

This Gaussian intensity model is known as single Gaussian background model. In the case of color frames, since pixel observation is a high dimension vector, $x_t \in \mathbb{R}^d$, a multivariate Gaussian density is considered, with a density given by

$$P(x_t|\mathscr{BG}) \approx \mathscr{N}(\mu, \Sigma) = \frac{1}{\sqrt{2\pi|\Sigma|}} \exp \frac{-1}{2}(x_t - \mu)^T \Sigma^{-1}(x_t - \mu), \tag{5.23}$$

where $\mu \in \mathbb{R}^d$ is the mean vector, and $\Sigma \in \mathbb{R}^{d\times d}$ is the covariance matrix of the distribution.

5.6.3 Mixture Gaussian Background Model

Typically, in outdoor environments with moving objects, the background of the scene is not totally static. For example, a pixel can present the image of the sky in a frame, a leaf of tree in another frame. In these cases, the pixel will have a different intensity. Therefore, a single Gaussian assumption for the probability density function of the pixel intensity will not be adequate. Rather et al. [21] proposed a generalization based on a mixture of Gaussians to model such variations. A mixture of three Gaussian distributions was used to model the pixel value of traffic surveillance applications. These three Gaussian distributions were corresponding to shadow, road, and vehicle distribution. Stauffer and Grimson [53] presented a generalized approach. A mixture of G Gaussian distributions (G is typically chosen between 3 and 5) is used to model the pixel intensity. The probability that a pixel has the intensity x_t at time t is given by

$$P(x_t|\mathscr{BG}) = \sum_{g=1}^{G} w_{g,t}\mathscr{N}(x_t; \mu_{g,t}, \Sigma_{g,t}), \tag{5.24}$$

where $\mathscr{N}(.; \mu_{g,t}, \Sigma_{g,t})$ is a Gaussian density function with a mean $\mu_{g,t}$ and covariance $\Sigma_{g,t} = \sigma_{g,t}^2 I$. $w_{g,t}$ is the weight for the g-th Gaussian components. The sum of weights across all components is equal to 1. The subscript t indicates that the mean, the covariance, and the weight of each component are updated at each step. The mixture model has proven to be very efficient both indoors and outdoors situations. Later, many modifications have been proposed to the Stauffer and Grimson's model [53]. An illustration of a real case of vehicle detection using background subtraction based on mixture models is given in Fig. 5.8.

5.6.4 Nonparametric Background Model

In the external scenes, there are dynamic zones such as the ripple of the waters. The modeling of these dynamic zones requires a more flexible representation of

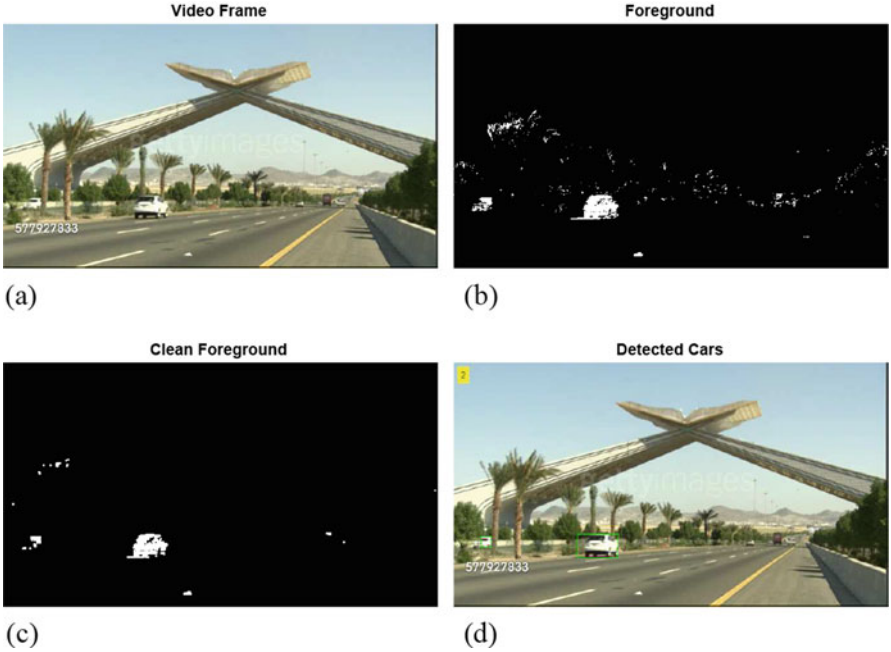


Fig. 5.8 Vehicle detection and counting in a video sequence using background subtraction based on Gaussian mixture models (GMMs). The video was taken from the entrance of Mecca (Makkah-Al-Mukarramah) city in Saudi Arabia (Source: Getty Images, Chicago, IL 60603 USA). (**a**) Arbitrary frame. (**b**) Noisy background. (**c**) Filtered background. (**d**) Detecting vehicles

the background probability distribution at each pixel. This encourages the use of a nonparametric density estimator for background modeling [19]. This technique consists of modeling the background with a kernel-based nonparametric model. The probability density of a pixel at time t is presented by the following equation:

$$P_k(X_t) = \frac{1}{N} \sum_{i=1}^{N} \phi(x_t - x_i) \tag{5.25}$$

where ϕ is the kernel function. Using a Gaussian-type kernel function, which amounts to considering a centered Gaussian function $\mathcal{N}(0, \Sigma)$, the density function is defined by:

$$P_k(X_t) = \sum_{i=1}^{N} \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left\{ -\frac{1}{2}(x_t - x_i)^T \Sigma_t^{-1}(x_t - x_i) \right\}. \tag{5.26}$$

For each pixel, the probability density is assumed $P_r(X_t)$. The pixel x_t is decided as a background pixel if it satisfies the following condition: $P_k(X_t) \geq \varepsilon$, with

ε a threshold to be arbitrarily fixed, and d represents the size of the space. It is notable that the method remains robust even in dynamic background frameworks with the presence of light perturbations. Nevertheless, introducing a kernel function significantly increases calculation time and complexity of the algorithm.

5.6.5 PCA-Based Background Subtraction

Oliver et al. [43] proposed a strategy based on a principal component analysis (PCA). The principle of the strategy is to have a background which describes the motion variation in a reduced multidimensional space. The PCA is known as a dimensionality reduction method. The authors suggested applying the ACP on N frames of the video to lead to a base of eigenvectors. Then, the first two eigenvectors are retained since they best explain the variance of these N frames, which can reduce the size of the space of representation. Once the model is built, each new frame I is projected onto the representation space in order to model the fixed regions of the scene. Objects are detected by calculating the difference between the input image I and the image I'. It is notable that this approach becomes ineffective if the background is evolutionary. The main steps of the approach called eigenbackgrounds are (Fig. 5.9):

- Arrange N frames into a column matrix M,
- Find the covariance matrix $C = MM^t$,
- Diagonalize the matrix C to obtain an eigenvector base ϕ and eigenvalues λ,
- Keep the two first eigenvectors,
- Project each new image I into the two-dimensional space, and then rebuild the image,
- Differentiate between the input image and the reconstructed image to detect the moving object.

5.7 Conclusion

Nowadays, modern digital traffic sensors and other intelligent surveillance technologies are increasingly deployed into transportation networks [45]. Infrastructure-based sensors are the most frequently used, seeing they are permanent and can be easily installed in the road or in its neighborhoods (e.g., on lighting poles, posts, buildings, and signs). They may be manually fixed during preventive road construction or maintenance, or by sensor injection machinery for rapid installation. Many other techniques like radar sensors, GPS, and digital cameras are increasingly used to measure important factors such as speed, category, and density of vehicles in order to provide traffic more reliable traffic information. Such a precious information will be subsequently exploited by planning and forecasting engineers to develop

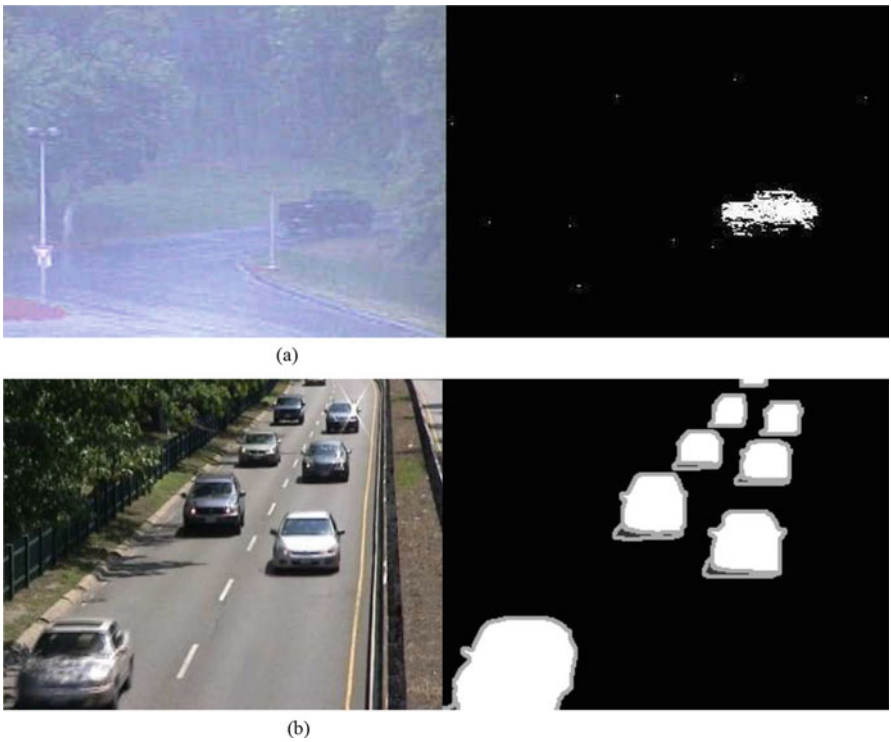


Fig. 5.9 Vehicle detection using (**a**) nonparametric background subtraction and (**b**) PCA-based background subtraction (Sources: (**a**) Elgamal et al. [19] and (**b**) Javed et al. [28])

intelligent interfaces. In this chapter, the main principles of traffic engineering, data collection, and measuring technologies as well as the most advanced vision-based models for the detection and tracking of vehicles are reviewed. In this context, we take stock of the recent literature on the main strategies for implementing these techniques with the aim of solving the major transportation problems, in particular, the congestion pollution.

Acknowledgements We would like to thank the anonymous reviewers for their insightful and constructive comments that have greatly contributed to improving the chapter and to the editorial staff for their generous support and assistance during the review process.

References

1. Ambardekar, A., Nicolescu, M., Bebis, G., Nicolescu, M.: Vehicle classification framework: a comparative study. EURASIP J. Image Video Process. **2014**(29), 1–13 (2014)
2. Anand, A., Ramadurai, G., Vanajakshi, L.: Data fusion-based traffic density estimation and prediction. J. Intell. Transp. Syst. **18**(4), 367–378 (2014)

3. Arnott, R.: On the optimal target curbside parking occupancy rate. Econ. Transp. **3**(2), 133–144 (2014)
4. Arnott, R., Kraus, M.: Congestion. In: Durlauf, S.N., Blume, L.E. (eds.) The New Palgrave Dictionary of Economics, 2nd edn. Palgrave Macmillan, London (2008)
5. Barcellos, P., Bouvié, C., Escouto, F.L., Scharcanski, J.: A novel video based system for detecting and counting vehicles at user-defined virtual loops. Expert Syst. Appl. **42**(4), 1845–1856 (2014)
6. Barria, J.A., Thajchayapong, S.: Detection and classification of traffic anomalies using microscopic traffic variables. IEEE Trans. Intell. Transp. Syst. **12**(3), 695–704 (2011)
7. Bas, E., Tekalp, M., Salman, F.S.: Automatic vehicle counting from video for traffic flow analysis. In: IEEE Symposium on Intelligent Vehicle - IV, pp. 392–397 (2007)
8. Bharadwaj, S., Ballare, S., Rohit, R., Chandel, M.K.: Impact of congestion on greenhouse gas emissions for road transport in Mumbai metropolitan region. In: Transportation Research Procedia, vol. 25, pp. 3538–3551 (2017)
9. Bouwmans, T., Porikli, F., Hörferlin, B., Vacavant, A.: Background Modeling and Foreground Detection for Video Surveillance: Traditional and Recent Approaches, Benchmarking and Evaluation. CRC Press, Taylor and Francis Group, Boca Raton (2014)
10. Button, K.: Transport Economics, 3rd edn. Edward Elgar, Cheltenham (2010)
11. Cambridge Systematics, Inc., Dowling Associates, Inc., System Metrics Group, Inc., Texas Transportation Institute: NCHRP Report 618: Cost-effective performance measures for travel time delay, variation, and reliability. Transportation Research Board, Washington, DC (2008)
12. Chang, Y.S., Lee, Y.J., Choi, S.S.B.: Is there more traffic congestion in larger cities? Scaling analysis of the 101 largest U.S. urban centers. Transp. Policy **59**, 54–63 (2017)
13. Chen, S., Shyu, M., Zhang, C., Strickrott, J.: A multimedia data mining framework: mining information from traffic video sequences. J. Intell. Inf. Syst. **19**(1), 61–77 (2002)
14. Chen, S., Zhang, J., Li, Y., Zhang, J.: A hierarchical model incorporating segmented regions and pixel descriptors for video background subtraction. IEEE Trans. Ind. Inf. **8**(1), 118–127 (2012)
15. Chilamkurti, N., Zeadally, S., Chaouchi, H.: Next-Generation Wireless Technologies: 4G and Beyond. Springer, London (2013)
16. Cohen, S.: Ingénierie du Trafic Routier: Elément de Théorie du Trafic et Applications. Eyrolles, Paris (1993)
17. Corréïa, A.: Modélisation de conflits dans l'algèbre des dioïdes - application à la régulation de trafic dans les carrefours, Thèse de Doctorat en Automatique, Université de Technologie de Belfort-Montbéliard, Besançon, France (2007)
18. de la Rocha, E., Palacios, R.: Image-processing algorithms for detecting and counting vehicles waiting at a traffic light. J. Electron. Imaging **19**(4), 043025-1–043025-8 (2010)
19. Elgammal, A., Harwood, D., Davis, L.: Non-parametric model for background subtraction. In: Proceedings of the 6th European Conference on Computer Vision-Part II, pp. 751–767 (2000)
20. Faouzi, N.E., Leung, H., Kurian, A.: Data fusion in intelligent transportation systems: progress and challenges – a survey. Inf. Fusion **12**(1), 4–10 (2011)
21. Friedman, N., Russell, S.: Image segmentation in video sequences: a probabilistic approach. In: Proceeding UAI'97 Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, pp. 175–181 (1997)
22. Fu-min, Z., Lü-chao, L., Xin-hua, J., Hong-tu, L.: An automatic recognition approach for traffic congestion states based on traffic video. J. Highw. Transp. Res. Dev. **8**(2), 72–80 (2014)
23. Hadi, R.A., Sulong, G., George, L.E.: Vehicle detection and tracking techniques: a concise review. Signal Image Process. Int. J. **5**(1), 1–12 (2014)
24. Hau, T.D.: Congestion pricing and road investment. In: Button, K.J., Verhoef, E.T. (eds.) Road Pricing, Traffic Congestion and the Environment, Chapter 3, pp. 39–78. Edward Elgar Publishing Limited, Cheltenham (1998)
25. He, S., Zhang, J., Cheng, Y., Wan, X., Ran, B.: Freeway multisensor data fusion approach integrating data from cell phone probes and fixed sensors. J. Sens. **2016**, Article ID 7269382, 13 pp. (2016)

26. Hymel, K.: Does traffic congestion reduce employment growth? J. Urban Econ. **65**(2), 127–135 (2009)
27. Jang, H., Won, I.-S., Jeong, D.-S.: Automatic vehicle detection and counting algorithm. Int. J. Comput. Sci. Netw. Secur. **14**(9), 99–102 (2014)
28. Javed, S., Oh, S.H., Heo, J., Jung, S.K.: Robust background subtraction via online robust PCA using image decomposition. In: International Conference on Research in Adaptive and Convergent System (ACM RACS 2014), pp. 105–110 (2014)
29. Jin, J., Rafferty, P.: Does congestion negatively affect income growth and employment growth? Empirical evidence from US metropolitan regions. Transp. Policy **55**, 1–8 (2017)
30. Klein, L.A., Mills, M.K., Gibson, D.R.P.: Traffic Detector Handbook, vol. I, 3rd edn. FHWA-HRT-06-108, FHWA, Washington, DC (2006)
31. Klein, R.W., Koeser, A.K., Hauer, R.J., Hansen, G., Escobedo, F.J.: Relationship between perceived and actual occupancy rates in urban settings. Urban For. Urban Green. **19**, 194–201 (2016)
32. Kolm, S.-Ch.: La théorie générale de l'encombrement [The General Theory of Congestion]. SEDEIS, Paris (1968)
33. Lagorio, A., Grosso, E., Tistarelli, M.: Automatic detection of adverse weather conditions in traffic scenes. In: IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance, pp. 273–279 (2008)
34. Laroche, F.: Économie politique des infrastructures ferroviaires. Thèse de Doctorat en Sciences économiques, Université Lumière Lyon 2, France (2014)
35. Levinson, H.S., Pratt, R.H., Bay, P.N., Douglas, G.B.: Quantifying Congestion, vol. 1, National Cooperative Highway Research Program (NCHRP): report 398. Transportation Research Board National Research Council, Washington (1997)
36. Lu, M., Sun, C., Zheng, S.: Congestion and pollution consequences of driving-to-school trips: a case study in Beijing. Transp. Res. Part D Transp. Environ. **50**, 280–291 (2017)
37. McCall, J.C., Trivedi, M.M.: Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. IEEE Trans. Intell. Transp. Syst. **7**(1), 20–37 (2006)
38. Mimbela, L.: A summary of vehicle detection and surveillance technologies used in intelligent transportation systems. Federal Highway Administration, Washington, DC (2007)
39. Minge, E.: Evaluation of non-intrusive technologies for traffic detection, Research Project, Minnesota Department of Transportation - Office of Traffic, Safety and Technology, Roseville, MN (2010)
40. Mohana, H.S., Ashwathakumar, M., Shivakumar, G.: Vehicle detection and counting by using real time traffic flux through differential technique and performance evaluation. In: International Conference on Advanced Computer Control, ICACC '09, pp. 791–795 (2008)
41. Narasimhan, S.G., Nayar, S.K.: Contrast restoration of weather degraded images. IEEE Trans. Pattern Anal. Mach. Intell. **25**(6), 713–724 (2003)
42. Narhe, M.C., Nagmode, M.S.: Vehicle counting using video image processing. Int. J. Comput. Technol. **1**(7), 358–362 (2014)
43. Oliver, N.M., Rosario, B., Pentland, A.: A Bayesian computer vision system for modeling human interactions. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 831–843 (2000)
44. Preethi, P., Ashalatha, R., Modelling saturation flow rate and right turn adjustment factor using area occupancy concept. Case Stud. Transp. Policy **6**(1), 63–71 (2018)
45. Rabbouch, H., Saâdaoui, F., Mraihi, R.: Unsupervised video summarization using cluster analysis for automatic vehicles counting and recognizing. Neurocomputing **260**, 157–173 (2017)
46. Raghtate, G., Tiwari, A.K.: Moving object counting in video signals. Int. J. Eng. Res. Gen. Sci. **2**(3), 415–420 (2014)
47. Rahane, S.K., Saharkar, U.R.: Traffic congestion – causes and solutions: a study of Talegaon Dabhade city. J. Inf. Knowl. Res. Civ. Eng. **3**(1), 160–163 (2014)
48. Réveillac, J.-M.: Modeling and Simulation of Logistics Flows 1: Theory and Fundamentals. Wiley, Hoboken (2017)

49. Robitaille, M., Nguyen, T.: Évaluation de la congestion "De la théorie à la pratique" Réseau routier de l'agglomération de Montréal, congrès annuel de 2003 de l'Association des transports du Canada à St. John's (2003)
50. Roess, R.P., Prassas, E.S., McShane, W.R.: Traffic Engineering, 3rd edn. Prentice Hall, Upper Saddle River (2004)
51. Shan, Z., Xia, Y., Hou, P., He, J.: Fusing incomplete multisensor heterogeneous data to estimate urban traffic. IEEE MultiMedia **23**(3), 56–63 (2016)
52. Shi, K., Di, B., Zhang, K., Feng, C., Svirchev, L.: Detrended cross-correlation analysis of urban traffic congestion and NO2 concentrations in Chengdu. Transp. Res. Part D Transp. Environ. **61**, 165–173 (2018)
53. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real time tracking. In: Computer Society Conference on Computer Vision and Pattern Recognition, Ft. Collins, vol. 2, pp. 246–252 (1999)
54. Texas A&M Transportation Institute (TTI): https://tti.tamu.edu/
55. Treiber, M., Kesting, A., Wilson, R.E.: Reconstructing the traffic state by fusion of heterogeneous data. Comput. Aided Civ. Infrastruct. Eng. **26**(6), 408–419 (2011)
56. Vickrey, W.: Congestion theory and transport investment. Am. Econ. Rev. **59**(2), 251–262 (1969)
57. Walton, C.M., Persad, K., Wang, Z., Svicarovich, K., Conway, A., Zhang, G.: Arterial Intelligent Transportation Systems – Infrastructure Elements and Traveler Information Requirements, Center for Transportation Research The University of Texas at Austin 3208 Red River Austin, TX 78705 (2009)
58. Wang, K., Yao, Y.: Video-based vehicle detection approach with data-driven adaptive neuro-fuzzy networks. Int. J. Pattern Recognit. Artif. Intell. **29**(7), 1–32 (2015)
59. Wardrop, J.G.: Some theoretical aspects of road traffic research. Proc. Inst. Civ. Eng. Part II **1**(36), 352–362 (1952)
60. Weibin, Z., Yong, Q., Zhuping, Z., Antonio, B.S., Minglei, S., Yinhai, W.: A method of speed data fusion based on Bayesian combination algorithm and Markov model. Transportation Research Board 97th Annual Meeting, Washington, DC (2018)
61. Wu, K., Chen, Y., Ma, J., Bai, S., Tang, X.: Traffic and emissions impact of congestion charging in the central Beijing urban area: a simulation analysis. Transp. Res. Part D Transp. Environ. **51**, 203–215 (2017)
62. Xia, Y., Shi, X., Song, G., Geng, Q., Liu, Y.: Towards improving quality of video-based vehicle counting method for traffic flow estimation. Signal Process. **120**, 672–681 (2016)

Chapter 6
How Did the Discussion Go: Discourse Act Classification in Social Media Conversations

Subhabrata Dutta, Tanmoy Chakraborty, and Dipankar Das

Abstract Over the last two decades, social media has emerged as almost an alternate world where people communicate with each other and express opinions about almost anything. This makes platforms like Facebook, Reddit, Twitter, Myspace, etc., a rich bank of heterogeneous data, primarily expressed via text but reflecting all textual and non-textual data that human interaction can produce. We propose a novel attention-based hierarchical LSTM model to classify discourse act sequences in social media conversations, aimed at mining data from online discussion using textual meanings beyond sentence level. The very uniqueness of the task is the complete categorization of possible pragmatic roles in informal textual discussions, contrary to extraction of question–answers, stance detection, or sarcasm identification which are very much role specific tasks. Early attempt was made on a Reddit discussion dataset. We train our model on the same data, and present test results on two different datasets, one from Reddit and one from Facebook. Our proposed model outperformed the previous one in terms of domain independence; without using platform-dependent structural features, our hierarchical LSTM with word relevance attention mechanism achieved $F1$-scores of 71% and 66%, respectively, to predict discourse roles of comments in Reddit and Facebook discussions. Efficiency of recurrent and convolutional architectures in order to learn discursive representation on the same task has been presented and analyzed, with different word and comment embedding schemes. Our attention mechanism enables us to inquire into relevance ordering of text segments according to their roles in discourse. We present a human annotator experiment to unveil important observations about modeling and data annotation. Equipped with our text-based discourse identification model, we inquire into how heterogeneous

S. Dutta (✉) · D. Das
Jadavpur University, Kolkata, India

T. Chakraborty
IIIT Delhi, Delhi, India
e-mail: tanmoy@iiitd.ac.in

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_6

non-textual features like location, time, leaning of information, etc. play their roles in characterizing online discussions on Facebook.

6.1 Introduction

While the predominant mode of people engaging in discussions on social media is via text, recent advents have pushed these communication to non-textual modes also. The most simple example of such communications are *reactions* in Facebook. People tend to express their opinions towards a content or opinions of others via categorized reactions like "love," "angry," or a mere simple "like." Platforms like Reddit provide "upvote" and "downvote" options to express categorized opinions. In an ongoing discussion, these options add newer structures of opinion exchange; users tend to express their opinion towards others using these options and such non-textual mode of discourse run parallel to the textual one.

There are still other factors effecting how online discussions proceed. There are heated topics of discussion where an user represents his/her community sentiments, like religion, race, political affiliation, location, etc. People bunk onto the sources of information like news reports, videos, and images, and these behaviors change with time. Even for a single topic, character of discussions varies temporally as new information floods in, from multiple modes. Exploring relationships between these heterogeneous features may reveal valuable understanding of online discussions. But to continue, one needs to identify intentions or roles of different people in discussion, depending on text only. Heterogeneity can be explored over that primary identification of discourse.

The term `Discourse` has been defined in numerous ways in linguist community. Broadly, discourse is how we meaningfully relate written or spoken natural language segments. In case of dialogues, we deal with compound discourses constituted by `Narrative Discourse` and `Repartee Discourse` [17]. While narrative discourse focuses on the depiction of motion, repartee discourse engages to describe speech exchanges. This second part varies in nature with varying types and platforms of dialogue. For example, in spoken dialogues, utterances of a single speaker are much more intervened compared to email conversation, due to more interruption and real-time transmission–reception when we talk face-to-face.

With the boom of social media, more and more people are expressing opinions, queries, and arguments on topics innumerable, opening a completely new type of repartee discourse. This has opened scopes of understanding how people engage in discussions. In fact, this can be extended to almost any platform where people interact with each other in an informal manner like Facebook, Twitter, Reddit, CreateDebate, etc. Participation in discussions is not homogeneous across these platforms; while Facebook or Twitter are more used for expression of opinions and argumentation, platforms like Reddit or different community forums have large usage for querying and answering. One method of understanding discussions has been to identify high-level discourse structures in such conversations. These

Table 6.1 Examples of discourse acts in discussion threads

Comment	Discourse act tag
U1: *I'm not form the US but am interested in US politics. So here is my question: Is Obamacare failing? If the democrats had won. Would they also be in a hurry to fix Obamacare because of flaws in the law?*	**Question**
U2 (replying **U1**): *The straight up answer to your question is "it's complicated". In my point, million have obtained insurance, some who were uninsurable before, primary care is available to more, maternal care, mental health, and more are available to closer to everyone in the US.*	**Answer**
U3 (replying to **U2**): *But not as many people obtain insurance as the CBO predicted in 2009, insurance costs too much, and the Obamacare is still a massive government overreach.*	**Disagreement**
U4 (replying to **U2**): *That was a good answer!*	**Appreciation**

structures tend to assign categories called `Discourse Acts` to each textual utterance (comments, messages, etc.) that pertain to their role in the conversation.

Like discourse processing in plain documents, discourse parsing of dialogues is a two-step process: identification of discourse constituent or elementary discourse units and then establishing discourse relation between them. In dialogues, each utterance is linked to the utterance it was replied to. In Table 6.1 **U2** is linked to **U1**, **U3** is linked to **U2**, and so on. In case of structured platforms like CreateDebate or Reddit, these links are already known. Each comment in these platforms is an explicit reply to some other. But in Facebook or various group chat platforms, this structure is not explicitly known. Dutta et al. [10] proposed a support vector machine-based framework to decide which comment is put in reply to whom, in case of Facebook discussion threads. A much complete work on tagging discourse acts of discussion comments is the `Coarse Discourse Dataset` [35]. This is a Reddit dataset with over 9000 discussion threads comprised of over 100,000 comments. Each comment is classified into one of the nine different discourse act tags, namely **Announcement**, **Question**, **Answer**, **Elaboration**, **Humor**, **Agreement**, **Disagreement**, **Appreciation**, and **Negative reaction**, with undecidable roles as **Other**.

Given the constituency information, a single discussion thread becomes a tree with the first or opening comment being the root node. Depth-first traversal of this tree yields multiple linear sequences of comments; each posed as a reply to its previous one. We then hypothesize that identification of discourse role of each comment depends on its ancestors along the chain and not only its parent comment; thus the problem becomes classification of sequence of comments to corresponding sequence of discourse acts. This is a familiar problem of mapping input sequences to same length output sequences, and a variety of traditional and neural learning models exist to solve this.

`Recurrent Neural Networks` or RNNs revolutionized the modeling of sequential data with its emergence. Given an RNN with x_t as input in current timestep, h_{t-1} as hidden layer output from previous timestep, output for current timestep o_t is computed as $f(x_t, h_{t-1})$ where the network learns function f.

Theoretically, this enables an RNN to learn dependencies in sequences which can be spread to arbitrary distances. Practically this is not the case, as *Vanishing Gradient Problem* [14] restricts RNNs to learn long-term dependencies. This is where `Long Short Term Memory` or LSTM models [13] come into play. In LSTMs, a separate memory cell is used to remember long-term dependencies, which can be updated depending on current input; so at each timestep, LSTM takes current input x_t and previous memory cell state c_{t-1} as input and compute output o_t and current cell state c_t. Governing equations of an LSTM are

$$i_t = \sigma_i(x_t W_{xi} + h_{t-1} W_{hi} + b_i) \tag{6.1}$$

$$f_t = \sigma_f(x_t W_{xf} + h_{t-1} W_{hf} + b_f) \tag{6.2}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \sigma_c(x_t W_{xc} + h_{t-1} W_{hc} + b_c) \tag{6.3}$$

$$o_t = \sigma_o(x_t W_{xo} + h_{t-1} W_{ho} + b_o) \tag{6.4}$$

$$h_t = o_t \odot \sigma_h(c_t) \tag{6.5}$$

where x_t is input vector, f_t is forget gate activation vector, i_t is input gate activation vector, o_t is output gate activation vector, h_t is output vector of LSTM, c_t is cell state, and W and b are weight and bias matrices. Malhotra et al. [20] showed that stacking up LSTM layers above each other enhances learning; deeper layers tend to model more complex representations from the previous layer output. This idea of stacking up multiple LSTM layers will be used in our experiments also.

LSTMs have been proved to be very much efficient in NLP tasks where sequence classification or time series forecasting applies [27, 28, 33]. LSTMs, when allowed to focus particular segments of input data, perform even better. Intuitively, this directs the model to remember relations regarding focused segments with additional priority. This is the basic idea behind `Attention Mechanisms`.

We pose our problem to identify discourse role of participants in online discussion with a broader problem of mapping heterogeneous real-world phenomenons with the characteristics of discussions on social media. We organize the rest of the chapter in the following manner:

- In Sect. 6.2 we present a survey of works related to linking real-world phenomenons with social media data, network analysis of social media interactions, discourse act tagging, and an important subproblem of stance detection.
- Section 6.3 presents the working and underlying rationale of our proposed multidimensional LSTM model with attention mechanism for discourse act tagging, along with one multi-layer perceptron model, two pure LSTM-based models, and one convolutional LSTM model.
- With the model definitions completed, we move on to experiment methodologies used in detail in Sect. 6.4.

- Section 6.5 contains the evaluations of the model performance along with a comparative study.
- We introspect into the shortcomings of our models and propose some possibilities of overcoming them in Sect. 6.6.
- In Sect. 6.7 we discuss how our model can be used to characterize online discussions to predict temporal variation of argumentation, possible effects of external sources of information, and emergence and reflection of community sentiments.

6.2 Related Work

Linking social media to real-world events has gained much focus from the start of this decade; mostly related to entity recognition and opinion mining. Bollen et al. [3] analyzed text contents of tweet streams to build a *mood time series* and studied its relation with the stock market time series. They used two different mood tracking tools to tag each tweet, tested those tools to predict moods of user regarding two different social events and finally devised a Granger causality analysis and a self-organizing fuzzy neural network to find correlation between this mood time series and daily up-downs of Dow Jones industrial average (DJIA) from March 2008 to December 2008. They achieved an accuracy of 86% to predict the direction of DJIA. Similar type of work was presented by Wang et al. [31]; they proposed a real-time system for Twitter sentiment analysis of US presidential election 2012. O'Connor et al. [23] linked several surveys on political presidential poll and consumer confidence in the USA to contemporary Twitter text sentiment. Most of these works and related ones tend to analyze sentiments of opinions expressed in social media data. Tan et al. [29] incorporated Twitter networking structure to classify user-level text sentiments.

As they become a platform to reflect thoughts and opinions, social media has its intrinsic role of networking, of connecting people and make some information flow. Subsequently, problems like community detection, information flow prediction, rumor detection, etc. emerge. Chakraborty et al. [5] presented a survey of metrics to evaluate community detection systems. Without going into much detail, one can refer to [19, 26, 32] as noteworthy works on analysis and prediction of what type of content propagates more over social media.

Research developments discussed till now have mostly relied on the semantics of the text in focus. Analyzing text in discourse level to do the same job is less explored yet promising approach. Trevithick and Clippinger [30] proposed how speech acts of message contents can be used to characterize relationship between participants in social networking. Somasundaran et al. [25] showed how opinion polarity classification can be improved by considering discourse relations.

The idea of discourse acts originated from spoken dialogues (also called `speech acts`). Bunt [4] proposed an ISO standardization for dialogue act annotation in spoken dialogues. Clark and Popescu-Belis [6] proposed MALTUS, a multi-layered discourse act tagging for spoken dialogues. In case of textual conversations

they do not readily translate. Repartee discourse, as discussed in Sect. 6.1, varies in qualitative nature when we go from speech to text, a major cause being the asynchronous nature of textual exchanges. Within the textual arena, formal communications like emails show distinctly different nature of exchange compared to informal conversations like chats or open discussion platforms. Assigning speech act like labels to emails was another approach [7]; this labeling was based on an idea about intention of action expressed by the mail sender, so they used tags like *request* or *commitment* as discourse roles. Kalchbrenner and Blunsom [15] proposed a neural model for classifying dialogue acts in transcribed telephone conversations. They used convolutional architecture to produce sentence representation from word embeddings, and then a recurrent network to map each convoluted utterance to corresponding discourse act. A similar type of architecture has been implemented in our work for comparison.

Most of the previous research in online discussion has focused on extraction of *question–answer* pairs. Ding et al. [8] developed a CRF-based model to identify context and question–answer discourse in a dataset constructed from Tripadvisor forum. Some endeavored on understanding argumentative discourse in online platforms dedicated for debating [12, 22] and limited to a handful of topics. Bhatia et al. [2] proposed a discourse act classification scheme on Ubuntu and Tripadvisor forum posts. Although they tend to classify not extract specific types of posts, the data they chose, and discourse acts they specified was more of query-solution type. Arguello and Shaffer [1] handled a similar problem, predicting discourse acts for MOOC forum posts. Both of the previous works focused on similar type of data on limited domain: they tend to classify discourse roles of posts which are put to mostly ask for help, answer some queries or evaluate some previous post. These are not like new age social networks with almost no bar on topics of discussion, covering from political debate to simple query-answering related to restaurants.

As mentioned earlier, a variety of approaches has been proposed regarding argumentation discourse, precisely, stance detection. Now stance detection can be viewed as a two-way problem. Both for monologues and dialogues, stance detection can be approached with fixed target set. That is, the subject about which to detect the stances is predefined. On the other hand, dynamic understanding of debate topic and stance detection around those topics is a much more complex task to solve. O'Connor et al. [23] presented a stance detection model from a large Tweet dataset covering numerous topics. Identification of political stances in social has gathered much focus recently. With a fixed target subject, Lai et al. [16] and Wang et al. [34] provide much insight to this problem.

The attention-based LSTM model proposed by Du et al. [9] bears much similarity with our proposed model. They addressed the problem of target specific stance detection using neural models. To make an LSTM focus on parts of text which may relate to the target subject, they used word vectors augmented with vectors representing words of target topic.

All these works mentioned earlier focused on specific types of discourse, and relied on data at par with those discourse types. To the best of our knowledge, Zhang et al. [35] proposed the first complete discourse categorization of textual

discussions in a broad platform, dealing with numerous topics. They proposed a CRF-based model to predict high-level discourse act labels of comments, using textual content-based features as well as structural features. With all the features, this model achieved a $F1$ score of **0.747**. But without structural features, $F1$ score dropped to **0.507**. The structural features they used were word counts in a comment, depth of comment in thread, length of sentences, etc. These features are highly dependent on which discussion forum is being referred to. Their dataset was prepared from Reddit discussion threads, and therefore these structural features predominantly correspond to Reddit's own discussion types. For example, if one plans to test this model on Twitter discussion, structural features will not translate due to word limit of tweets. Same goes for Facebook as users idea about engaging in conversations in Facebook differs from Reddit, thereby changing the way people talk. Scott et al. [24] and Misra et al. [11] presented two independent studies on discourse of social media, focusing on Twitter and Facebook, respectively. Both of them show a common finding, mediator platform with its functionality and constraints largely determines the pragmatics of mediated conversations. That is why we use the coarse discourse dataset as primary data to train and test our models, but exclude explicit representation of structural features and focus solely on content to make our model platform independent.

6.3 Model Description

As there is no previous neural network model for the high-level discourse labeling tasks in case of asynchronous textual conversations, to the best of our knowledge, we devised **five** different neural network-based models along with our finally proposed multidimensional LSTM model with word relevance attention mechanism. We begin by describing these models.

6.3.1 Multi-Layer Perceptron Model

In Sect. 6.1 we hypothesized that prediction of discourse role of a comment in a chain is affected by all its ancestors in that chain. An MLP model contradicts this hypothesis, predicting discourse acts one by one. We present this model to justify our hypothesis.

Let C_i and C_{i+1} be the pretrained vector representation of comments in a chain where C_{i+1} is put in reply to C_i; D_i is the discourse act represented in an one-hot vector of size 10 (no. of classes). Input to our MLP is a vector I resulting from concatenation of C_i, D_i, and C_{i+1}, and model predicts $D_i + 1$. Stepwise internal computations are as follows:

$$\mathbf{H_1} = \mathbf{f_1}(\mathbf{I} \bullet \mathbf{W_1} + \mathbf{B_1}) \tag{6.6}$$

$$\mathbf{H_2} = \mathbf{f_1}(\mathbf{H_1} \bullet \mathbf{W_2} + \mathbf{B_2}) \tag{6.7}$$

$$\mathbf{D} = \mathbf{f_2}(\mathbf{H_1} \bullet \mathbf{W_2} + \mathbf{B_2}) \tag{6.8}$$

where f_1 and f_2 are *Sigmoid* and *Softmax* nonlinearity, respectively, **W** and **B** are weight and bias matrices of corresponding layer. Last output **D** gives a probability distribution over the 10 classes to predict. The MLP tries to minimize the *categorical cross entropy* of **D** in the course of learning.

6.3.2 LSTM with Pretrained Comment Vectors

Now to go along with our hypothesis, we design a simple LSTM model (Fig. 6.1) with sequence of comment vectors $\mathbf{C} = [C_0, C_1, \ldots, C_n]$ as input and sequence of predicted discourse acts $\mathbf{D} = [D_0, D_1, \ldots, D_n]$ as output. Sequential outputs from the LSTM are connected to a densely connected layer with softmax activation to get the probability distribution of D_i's, just like Eq. (6.8). Similar to the MLP, this model attempts to minimize *categorical cross entropy* for each $D_i \in \mathbf{D}$.

6.3.3 Two-Dimensional LSTM

Previous two models take each comment as a pretrained vector. Our next model explores the task word by word and representations of comments are learned within discourse act prediction task. Rationale behind this was the assumption that, given the discourse act tagging target, intermediate representation of comments learned

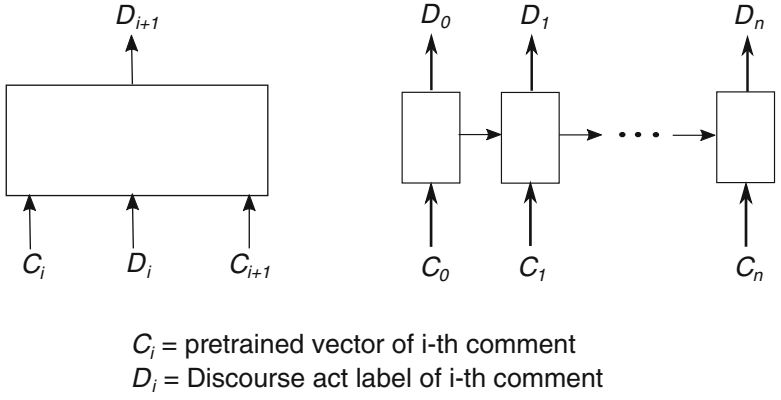


Fig. 6.1 Architectures of MLP (left) and LSTM (right) using pretrained comment vectors

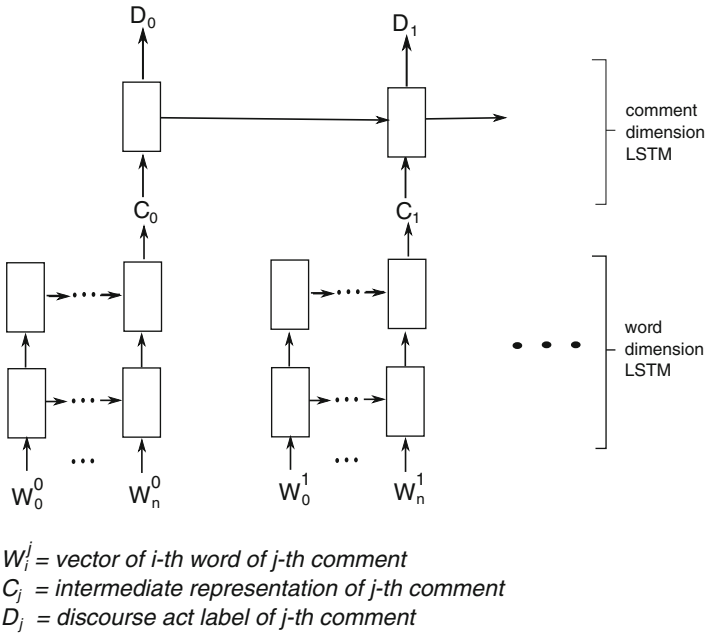


Fig. 6.2 Architecture two-dimensional LSTM

by sequential processing of words will contain more long distance dependencies between words. As we are dealing with pragmatic task, larger contexts of words are needed to be taken into account, and theoretically LSTMs can capture such dependencies better. As depicted in Fig. 6.2, each comment C_j is represented as a sequence of words $\{W_i^j\}$. The model can be segmented into two parts: *word-dimension stacked LSTM* and *comment-dimension LSTM*. If we denote each LSTM layer as a black-box function $\mathscr{L}$, then the stepwise computations are as follows:

$$\mathbf{W}' = \mathscr{L}(\mathbf{W}) \tag{6.9}$$

$$\mathbf{C} = \mathscr{L}'(\mathbf{W}') \tag{6.10}$$

$$\mathbf{D} = \mathscr{L}(\mathbf{C}) \tag{6.11}$$

where $\mathscr{L}$ returns sequential output and $\mathscr{L}'$ returns single output.

We used one-hot vector representation for the input words; an embedding layer was used to produce distributed word vectors. Weights of this embedding layer was set to pretrained word vectors, so that during training, our model fine-tunes the word vectors.

6.3.4 Two-Dimensional LSTM with Word Relevance Attention

Compared to the MLP model, those using LSTMs approach the problem of discourse act prediction task as a sequence-to-sequence modeling. In two-dimensional LSTM model, we have taken into account how the comments are being constructed from sequence of words to capture pragmatic relation between words. Still, in two-dimensional LSTM model we just discussed, intermediate representation of each comment depends only on the words constituting itself. This bars the model to pick relevant words in relation to the previous comments, topics of discussion, etc., and simply same words in two different comments with two different discourse context get same relevance. We propose an attention mechanism with the two-dimensional LSTM model to let it learn to assign more relevance on particular words depending on context.

6.3.4.1 Word Relevance Attention

Apparently question–answer discourse is easy to identify. Questions possess distinct parts-of-speech ordering (mostly starting with *verbs*), and answers are always paired with questions. Discourse roles like agreement, disagreement, humor, etc. are rather complex to distinguish. These discourse relations can often be identified with stances and content to justify stances. These stances can be based on the topic of discussion or the parent comment. That is, the task becomes to identify stance of two comments against a set of common topic words. We exploit a special structural leverage of online public forum discussion to identify topic of discussion. In case of human–human interaction via speech or via personal messages, usually a person is able to recall only a last few utterances. But in public forums, all the previous comments are open to read. Out of all those messages, the first (thread starter) comment is the one defining the topic of discussion. Every user tries to post their comment in relevance to those topics.

The attention mechanism we propose is hypothesized to exploit these phenomenons. We select specific words (*nouns*, *verbs*, *adjectives*, and *adverbs*) from the first comment of the thread and the parent comment (one that is replied to) as *discourse target*. We take weighted mean of these words to produce two vectors, corresponding to the first comment and the previous comment (in our experiment we used *tf-idf* weights to focus on important words from these two comments). These two vectors are then augmented with each word of the current comment. If $\mathbf{W_f} = [w_0^f, \ldots, w_{M-1}^f]$ and $\mathbf{W_p} = [w_0^p, \ldots, w_{N-1}^p]$ be the sequence of words extracted from the first and the previous comments, respectively, then

$$W'_f = \frac{\sum_{i=0}^{M-1} w_i^f t_i^f}{M} \tag{6.12}$$

$$W'_p = \frac{\sum_{i=0}^{N-1} w_i^p t_i^p}{N} \tag{6.13}$$

where t_i^f and t_i^p represent tf-idf value of the i-th word in $\mathbf{W_f}$ and $\mathbf{W_p}$, respectively. We do not use plain tf-idf values calculated with comments being represented as documents. Instead, as proposed in [10], we use hierarchical frequencies. As the first comment represents topic of discussion, and this is a characteristic of the thread, we take threads as documents and calculate *inverse thread frequency* of the words. With the previous comment words, we rather focus on the discussion context, which changes comment by comment. To extract relevant context words, for the previous comment, we compute *inverse comment frequencies* over the whole dataset. So t_i^f is term frequency multiplied by inverse thread frequency, whereas t_i^p is term frequency multiplied by inverse comment frequency.

W'_f and W'_p now can be visualized as vectors representing relevant contents of first and previous comment, respectively. They are concatenated with each word vector of the current comment to generate target augmented vectors $\mathbf{T} = [T_0, \ldots, T_C]$ where C is the number of words in current comment (Fig. 6.3).

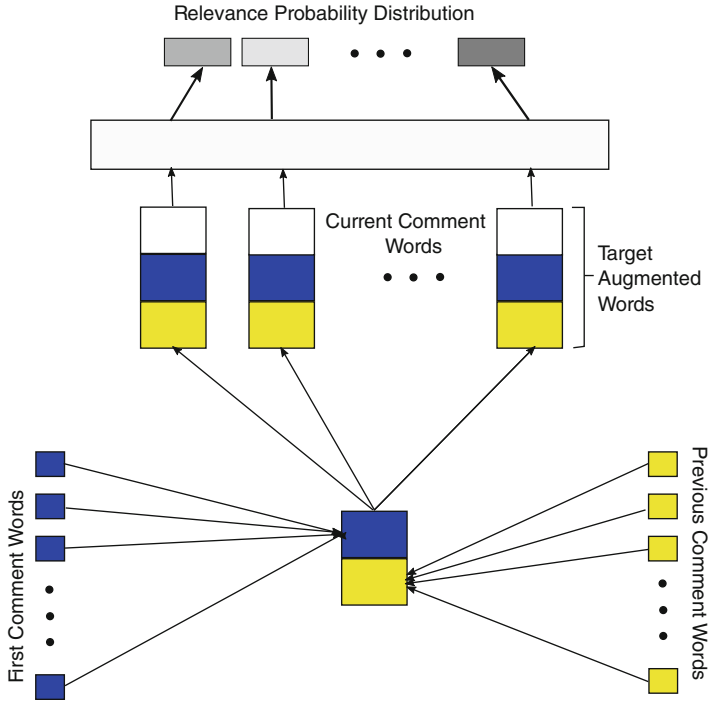


Fig. 6.3 Word relevance attention architecture

Now we are going to make our model learn relevance ordering from **T**. Given dimensionality of the word vectors used V, we initialize a trainable weight matrix **K** of order $C \times 3V$, and produce a vector $S = [s_0, \ldots, s_C]$ such that,

$$s_i = \sum_{j=0}^{3V} \mathbf{K}[i][j] * \mathbf{T}[i][j] \qquad \forall i \in [0, C] \tag{6.14}$$

This operation serves two purposes. Suppose, $\mathbf{K}[i]$ contains all ones and all the word vector entries of $\mathbf{T}[i]$ are positive. Then larger values of s_i will signify greater similarity between target word vector and i-th word vector of current comment. Practically word vector entries will be both positive and negative, and the attention may learn corresponding weight values in **K** to maximize s_i for similar words. On the other hand, **K** may learn to pose proper weights over related adjectives or negation words and even discourse connective prepositions to reflect their relevance when generating comment vectors. Applying *softmax* over S, we get a probability distribution $P = [p_0, \ldots, p_C]$ over the words of current comment.

Now we get back to two-dimensional LSTM model discussed in Sect. 6.3.3. The relevance probability P is applied to the outputs of the first LSTM of the word-dimension part,

$$\mathbf{W_{rel}}[i] = p_i \mathbf{W}'[i] \qquad \forall i \in [0, C] \tag{6.15}$$

where $\mathbf{W}'$ corresponds from Eq. (6.9) and $\mathbf{W_{rel}}$ is the focused word sequence representation which will be fed to the next LSTM layer to generate comment vector.

6.3.5 *Convolutional Generation of Intermediate Comment Vectors*

In our proposed two-dimensional LSTM model, intermediate representation of each comment was computed using a stacked LSTM model. To compare the performance of this model, we also devised a convolutional model which generates a single vector from the *word images* (sequence of words now constituting a 2D matrix of size $C \times V$) by convoluting and average-pooling. This model actually resembles to an N-gram model, as we use three parallel 1D convolution with filter sizes two, three, and four and concatenate the three pooling results to a single vector representing the comment. So the model actually learns to extract bi-gram, tri-gram, and 4-gram features from the comments to generate the vector. Convoluted and concatenated vector is then fed to an LSTM to predict the discourse act sequence, identical to the comment-dimension LSTM of Sect. 6.3.3. This model is experimented both with and without the attention mechanism proposed (Fig. 6.4).

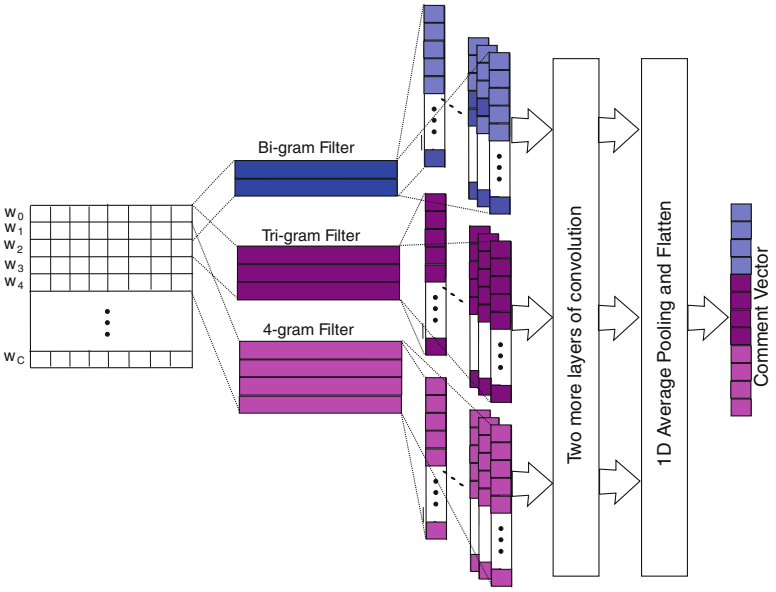


Fig. 6.4 Architecture of convolutional generation of comment vectors (output goes to LSTM)

Table 6.2 The presence of different discourse act classes in coarse discourse dataset

Discourse act	Total no. of comments	% in dataset
Question	17,681	17.6
Answer	41,658	41.5
Announcement	2024	2.0
Elaboration	18,927	18.8
Agreement	5072	5.1
Disagreement	3436	3.4
Humor	2409	2.4
Appreciation	8807	8.8
Negative reaction	1899	1.9

6.4 Experiments

6.4.1 Dataset

We used the *course discourse dataset*[1] of Reddit discussions as our primary dataset to train and test our models. An in-depth analysis of the data can be found in [35] (Table 6.2).

[1] https://github.com/dmorr-google/coarse-discourse.

Another dataset of comparably smaller size was also used, solely to test the best three models from test results on the coarse discourse dataset. This dataset is an extended version of the Facebook discussion dataset prepared by Dutta et al. [10]. We manually annotated the data with discourse act labels as per the rating guidelines with the coarse discourse data. This dataset contains 20 threads of discussions under posts from Facebook pages of various newsgroups like BBC, The New York Times, The Times of India, The Guardian, etc., with a total of 1177 comments altogether. These discussions were mostly related to heated contemporary social and political issues like *US Presidential Election 2016*, *Terrorist Attacks*, *Brexit*, etc.

6.4.2 Pretrained Embeddings

As discussed in the model descriptions, we used either pretrained word or comment embeddings. For word embeddings, we used Word2Vec [21] on the comments from both the datasets taken together. We generated six different word embedding sets with vector size 50, 100, 150, 200, 300, and 400 and ran pilot experiments with the two-dimensional LSTM and convolutional LSTM models using each embeddings on a small fraction of the dataset. We observed that increasing vector size beyond 150 did not bring any further betterment of performance and only burdened the hardware performance. So we stick to the 150-length word embedding throughout the experiments.

Same idea was used for pretrained comment embeddings. Doc2Vec [18] was used to produce ten different paragraph embeddings ranging from size 500 to 1500, and finally the one with size 700 was taken.

6.4.3 Training Models

We used fivefold stratified cross validation to train and test all the models. As the length of comment sequences varies from 2 to 11, and further number of words present in each comment varies substantially, we had to pad each comment with zeros. Longer chains are actually rarer, and thus to avoid too much padded data, we take equal length sequences at a time. That is, we train a model on sequences of length i, save the learned weights, and load them to a new model for sequences of length $i + 1$. We optimized hyper-parameters of all the models using scikit-learn's[2] grid search.

A CRF model using all the features (content + structure) used by Zhang et al. [35] was also trained. We test this model on Facebook dataset to present a full comparison of our models along with the state of the art.

[2] http://scikit-learn.org.

6.4.4 Human Annotator Testing

From the *inter-annotator reliability* measures presented by Zhang et al. [35], one can see that distinguishing discourse act labels other than question, answer, and announcement is a very much subjective process. So we devised an evaluation of model performance by human subjects from linguistic and non-linguistic backgrounds. A small fraction of the testing data from Reddit dataset was used for this purpose (20 threads containing a total 657 sequences), with the same rating guidelines. Two different set of experiments were done using each group of annotators:

1. **Zero knowledge testing of model**. Annotators are presented with predictions made by a model. They evaluate the predictions without prior knowledge of the labeling in corpus.
2. **Zero knowledge testing followed by allowed correction**. Annotators do zero knowledge testing, then we present them the corresponding labeling in corpus. They are allowed to make changes in their previous decisions if they think so.
3. **Self-prediction**. Annotators were asked to predict discourse tags, and we evaluated their performance compared to the corpus labels.

6.5 Observations

6.5.1 Model Performances

We start with presenting results of testing all the models proposed on coarse discourse dataset. As shown in Table 6.3, our first hypothesis of modeling the problem as a seq2seq prediction task clearly holds true. All four models processing sequence of comments as input outperforms the MLP model by big margin. We do not use this model for further experiments.

In an overall performance, convolutional generation of comment representation had an edge over recurrent generation when not equipped with the word relevance attention. To dig a bit further, we need to look into the class-wise performance of the

Table 6.3 Overall performance of all models tested on Reddit dataset

Models	Precision	Recall	$F1$ score
MLP	0.51	0.43	0.46
LSTM with paragraph-vectors	0.58	0.57	0.57
2D LSTM	0.60	0.61	0.60
CNN-LSTM	0.62	0.60	0.61
CNN-LSTM with attention	0.65	0.63	0.64
2D LSTM with attention	0.72	0.71	0.71

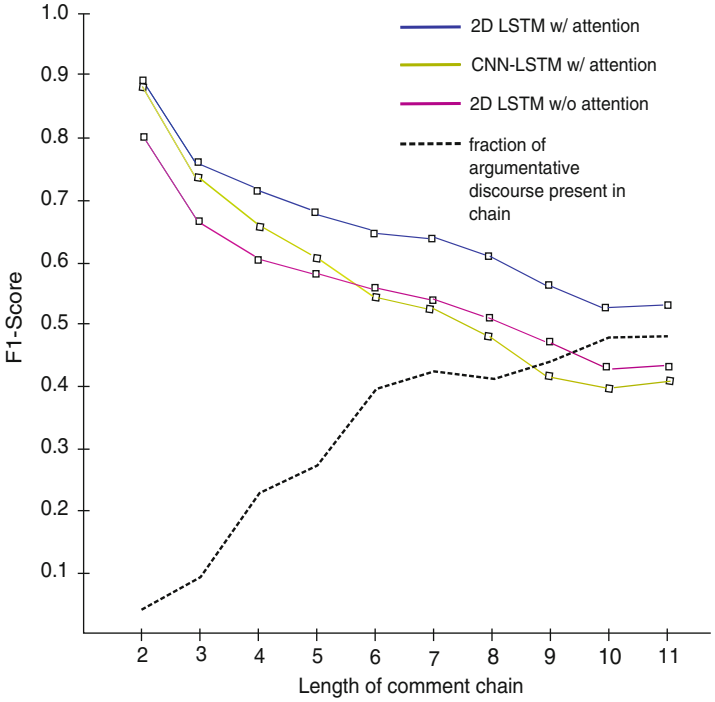


Fig. 6.5 Model performances and fraction of argumentative discourse roles present in sequence vs. length of comment sequence

four models presented in Fig. 6.6. The CNN-LSTM model actually performed much better compared to the 2D LSTM in classes like question, answer, announcement, and elaboration. But classes like humor, agreement, disagreement, appreciation, and negative reaction show the other way round. Though this phenomenon demands an in-depth analysis, we present a rather intuitive explanation here.

Our CNN-LSTM model takes bi-grams, tri-grams, and four-grams from the comment consecutively and constructs parallel representations based on them. This captures the local organization of text very well. Discourse roles of the first type are more easily predictable with such local organizations. But they fail to capture long distance relationships between words in a text, which is more important to identify argumentative discourses of the second type. LSTMs have an edge over CNNs when the problem is to identify long dependencies.

But simple 2D LSTMs does not learn to know which long-term dependencies are actually to put more focus on. Word-dimension part of the model tries to capture internal discourse organization of a comment without taking into account what the high-level discourse is going on. Theoretically, making these LSTMs stateful would have solved this problem, but in reality mere statefulness cannot capture this much information content. The attention mechanism we proposed tried to solve

this problem, at least partially. We can see the sheer rise of performance in both the CNN-LSTM and 2D-LSTM model when equipped with the word relevance probability computed using previous and first comment words. But here again, 2D LSTMs exploited the attention much more rigorously compared to the convolutional one.

From the nine discourse act roles, we identified *agreement*, *disagreement*, *humor*, *appreciation*, and *negative-reaction* as part of argumentative discourse. In Fig. 6.5, the presence of argumentative discourse can be seen more manifesting in longer chains. As the length of chain increases, two challenges occur simultaneously: the comment-dimension LSTM has to remember the learnings from more previous comments; on the other hand, with the increase in argumentative discourse, stance detection problem with dynamic targets comes more into play. We can check the performances of our two-dimensional LSTM model, with and without word relevance attention, and CNN-LSTM with word relevance attention, given the variation of the two challenges (Fig. 6.6).

Now we move on to compare our best two models (CNN-LSTM and 2D LSTM, both with attention) with the state-of-the-art CRF-based model by Zhang et al. [35]. On the Reddit dataset, CRF model with only content features (with $F1$ score 0.50)

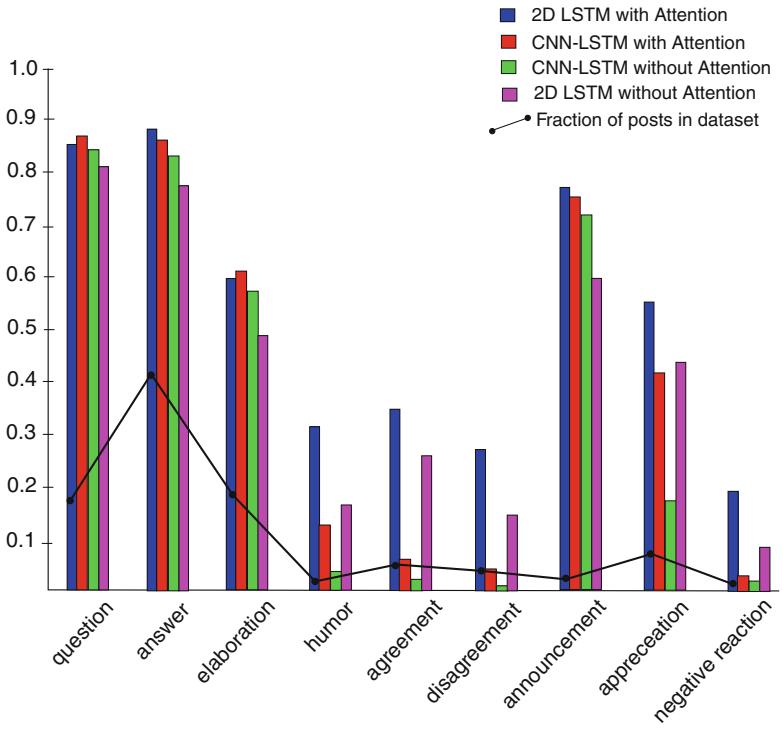


Fig. 6.6 Class-wise $F1$-scores of the four best performing models

was clearly outperformed by our models by big margin. However, the one with all the features still got better results compared to us, with $F1$ score 0.74. But on the Facebook dataset, as presented in Table 6.4, our 2D LSTM with word relevance attention clearly performed better compared to the all feature CRF. Though this test is not exhaustive with data from other discussion platforms, we can still conclude that our model achieved better domain independence.

6.5.2 Human Annotator Evaluation

Low inter-annotator reliability presented for the Reddit dataset reflects substantially in the test results from human evaluation. As we can see, when presented with the annotations done by our 2D LSTM with attention model, human annotators evaluated those to be better performing compared to the actual results evaluated by corpus annotations. When presented with the corpus annotations, these annotators changed their decision to decrease the scores. As we can see, this change is much higher for annotators from non-linguistic backgrounds (Table 6.5).

When annotators were asked to predict the labels themselves, their performance also varied by a big margin. Average scores for linguistic and non-linguistic background annotators have been presented. Worst case to best case results varied from 0.68 (by a non-linguistic annotator) to 0.82 (by a linguistic annotator). We also observed them to tag same comment in different sequences with different discourse roles.

Table 6.4 Comparison with state-of-the-art model on Facebook data

Models	Precision	Recall	$F1$ score
CNN-LSTM with attention	0.61	0.57	0.58
CRF with all features	0.62	0.63	**0.62**
2D LSTM with attention	0.65	0.68	**0.66**

Bold values highlight the previous state-of-the-art and development we made

Table 6.5 Results of human annotator testing

Annotator background	Evaluation method	$F1$ score
Linguistics	Zero knowledge	0.74
	Zero knowledge with correction	0.72
	Self-prediction	0.81
Non-linguistics	Zero knowledge	0.77
	Zero knowledge with correction	0.71
	Self-prediction	0.73
Actual performance		0.70

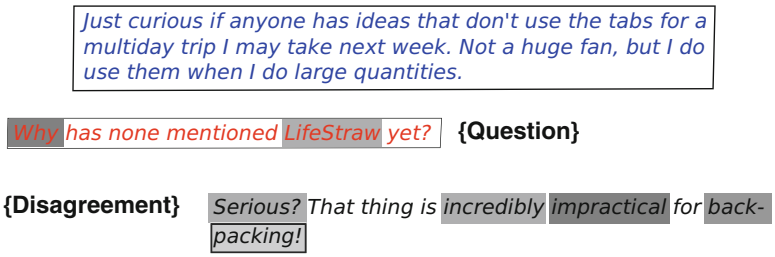


Fig. 6.7 Example of word relevance learned; blue text denotes first comment; darker labels on words signify higher relevance assigned

6.5.3 Learning Word Relevance

Figure 6.7 shows an example of how the attention mechanism weighted different words in the comment depending on the first and the previous comment. The comment shown in example acquired a disagreement-type discourse role. Darker gray labels signify higher relevance values assigned to the word by the attention mechanism. From the first comment we can see that the discussion evolves around some backpacking tools. The parent comment mentioned a named entity, possibly a particular brand name. The attention framework picked up the adjectives and adverbs in the current comment to as more relevant words. Also a discourse connective at the beginning was identified to mark the discourse role. For the parent comment also, words identifying the query nature of the comment have been put on more weighting.

From this relevance ordering, two related observations can be made. Firstly, with more fine tuning and a lot more of training data, this framework can lead us to an unsupervised marking of text for discourse connectives, importantly, high-level discourse connectives. Unlike discourse connectives present in a single document, dialogue cues marking high-level discourse might be expressed by word structures like some tree or graph. They do bear non-consecutive long distance interdependencies for sure. Secondly, for argumentative/stance-taking discourse, word relevance resembles sentiment lexicons.

6.6 Error Analysis and Possible Solutions

Results of human annotator testing clearly indicate that labeling of discourse roles is very much a subjective process. In fact, a comment in a thread can have different discourse roles when taken in different chain of linked comments. Comments which manifest multiple discourse roles (e.g., disagreement and negative reaction simultaneously) should be allowed to be labeled with multiple tags with possibly a

probability distribution over them. Or we can devise hierarchical labeling, so that each comment has a broad level discourse act tag, and more sub-acts under that.

We devised our attention mechanism to make the word-dimension LSTM peep into words from first and parent comment. But our very assumption that discourse act tagging is a seq2seq problem is not actually being exploited here. The flow of narrative discourse within comments in the chain other than the first and the parent one must be taken into account when the sequence of words is being modeled into a comment vector.

With the attention mechanism, our model was able to distinguish between query-type discourse and argumentative discourse. But within these broad discourse types, every discourse classes bear some more complex semantic and pragmatic features, and a further focus is necessary for better prediction of similar discourse roles. For example, our model misclassified only 7% of comments of query-type discourse as an argumentative one. But a huge 22% of negative reaction-type comments have been misclassified as disagreement. Same goes for other similar classes. If we take disagreement vs negative reaction confusion as an example, focusing on textual features regarding justification or subjectivity would help us more. Humors are actually the most complex discourse to predict. In an argument, without the knowledge of the stance of a person beforehand, humors (to be precise, sarcasms) are almost similar to agreements or appreciations. Thus a speaker profiling would also be needed for the task. Coarse discourse dataset has an average 1.66 posts from each author. This is pretty high for a single dataset but not enough for a complex task like speaker profiling, specially using neural models. Possibly a joint source model can help here, where a separate author data will be collected and used in parallel.

Using pretrained word vectors incorporate semantic relations of words in a comment. For example, this makes the model to identify *Democrats* being related to *politics* or *republicans* or *Hilary*. But this can never capture the antagonistic relation between *Obamacare* and *Trump*. Thus, given two short texts with such words and not much explanation (discourse roles like negative reaction), a model would need either much larger training set or some way to incorporate world knowledge. With our model, many of such instances have been misclassified.

Last but not the least, though we took a linear sequence of comments as input to classify, almost all discussion platforms originally follow a tree structure (the one from which we picked up each individual sequence). Now when a person starts typing a reply to a certain comment, he or she can see all the other comments present there; possibly many of those comments are reply to the comment he or she is writing a reply to. And these comments do have weak effects on discourse roles; that is, not only the linear sequence, the whole thread tree needs to process at a time for better understanding of the discourse.

6.7 Characterizing Discussions with Discourse Roles

Up until this point, we discussed our proposed model to classify discourse acts of comments from text. With our nine different discourse roles, we can, at least apparently, characterize discussions and explore temporal patterns, community sentiments, etc. We separately collected 12 threads of discussion regarding *Jallikattu* comprising 894 comments from 59 users. These 12 threads are the discussions on Facebook pages of different news groups posting reports and videos about Jallikattu. We used our hierarchical LSTM with attention to tag each comment with discourse roles and explore the following hypothesis:

1. A set of threads with more question–answer or appreciation tags can be hypothesized to be one where participants are acquiring and sharing information. On the other hand, threads with disagreement, negative reaction, or humor tags are more likely to be argumentative or quarrelsome one. Temporal changes in nature of discourse reflect changing engagement of people.
2. Disagreement, negative reaction, or humor relation between two comments group the corresponding users to antagonistic communities, whereas agreement and appreciation reflect a belonging to similar community.

Jallikattu is an ancient ritual in Tamil Nadu (southern-most state of India), much like bull fighting. In 2016, a massive unrest followed the government decision to ban the festival. The decision was taken on 14th January. Figure 6.8 shows the change in discourse type in discussions over Facebook as time goes. One can clearly observe that, at the beginning the discourse showed an inquisitive nature, with mostly question–answer-appreciation type discourse acts prevailing. One explanation can be, within this time, people engaging in discussions were not much aware about

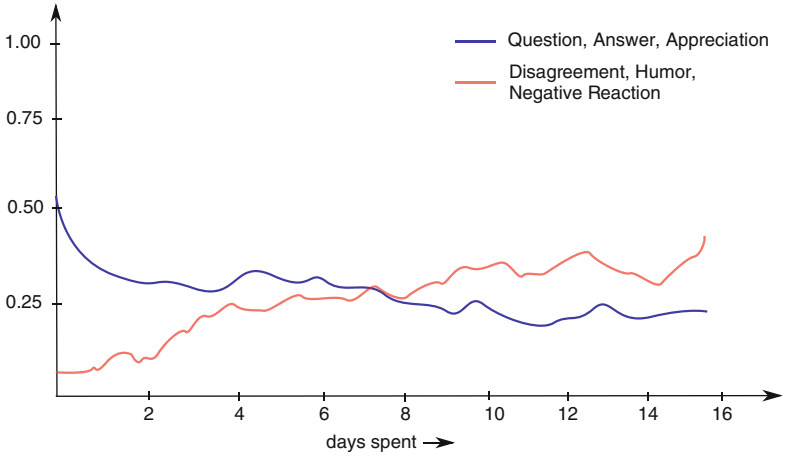


Fig. 6.8 Temporal plot showing how fraction of discourse patterns change with time spent over Jallikattu debate

what is happening and the focus was to know things. As time passes, mode of engagement changes more towards argumentation, with people taking firm position.

Our attempt to explore our second hypothesis revealed some actual community patterns. As stated, we tried to group people into two partitions depending on the discourse acts of their comments and to whom these were posted. For this purpose, we manually collected geographical location of the users participating in those discussions. Our observation was quite aligned with the real-world situation: users almost got partitioned into two antagonistic groups, one mostly comprised of users from Tamil Nadu, and the other mostly from northern India.

A peculiar pattern about nature of argumentation revealed, depending on the original post under which the discussion is going on. When the original post was video or images of unrest, we got a sharp rise in two specific discourse acts, negative reaction and appreciation. In case of original post being extensive news report or socio-cultural analysis, elaboration, agreement, and disagreement prevailed. The former can be specified as a case of subjective argumentation, with less objective reasoning, while the latter may contain factual argumentation.

6.8 Conclusion

Understanding complex pragmatics is a new and intriguing problem posed in front of computational linguists. With dialogues, this becomes far more challenging. In those work, we present a neural model which, naively speaking, employs layered memory to understand how individual words tend to constitute a meaningful comment, and then how multiple comments constitute a meaningful discussion. With the word relevance attention mechanism, our model tends to learn which words should be given more importance while deciding the discourse role of a comment. The proposed multidimensional LSTM with word attention not only outperformed the previous work on complete discourse act tagging, but also yielded better results in subproblems like question–answer extraction or stance detection compared to many previous works.

We can extend this idea of world relevance to larger contexts than only the previous and first comments. This may even equip us with an introspection into propagation of topic, sentiment, and other linguistic features through conversations. We used the only available dataset with a sizable amount of training samples, and the imbalance of per-class samples might have restricted our model to achieve best performance. But with sufficient amount of training data from different platforms, this model can possibly present a breakthrough in analysis of discourse in dialogues.

We used our model to characterize a small stream of discussions regarding a single topic; that too in a single platform. But this revealed the potential of discourse relation to be exploited in social network analyses. This can be further extended to understand argumentation, particularly in multimodal environment, and how different information sources tend to shape online discussions.

References

1. Arguello, J., Shaffer, K.: Predicting speech acts in MOOC forum posts. In: ICWSM, pp. 2–11 (2015)
2. Bhatia, S., Biyani, P., Mitra, P.: Identifying the role of individual user messages in an online discussion and its use in thread retrieval. J. Assoc. Inf. Sci. Technol. **67**(2), 276–288 (2016)
3. Bollen, J., Mao, H., Zeng, X.: Twitter mood predicts the stock market. J. Comput. Sci. **2**(1), 1–8 (2011)
4. Bunt, H.: A methodology for designing semantic annotation languages exploring semantic-syntactic isomorphisms. In: Proceedings of the Second International Conference on Global Interoperability for Language Resources (ICGL 2010), Hong Kong, pp. 29–46 (2010)
5. Chakraborty, T., Dalmia, A., Mukherjee, A., Ganguly, N.: Metrics for community analysis: a survey. ACM Comput. Surv. **50**(4), 54 (2017)
6. Clark, A., Popescu-Belis, A.: Multi-level dialogue act tags. In: SIGDIAL, Cambridge, pp. 163–170 (2004)
7. Cohen, W.W., Carvalho, V.R., Mitchell, T.M.: Learning to classify email into "speech acts". In: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pp. 1–8 (2004)
8. Ding, S., Cong, G., Lin, C.Y., Zhu, X.: Using conditional random fields to extract contexts and answers of questions from online forums. In: Proceedings of ACL-08: HLT, pp. 710–718 (2008)
9. Du, J., Xu, R., He, Y., Gui, L.: Stance classification with target-specific neural attention networks. In: International Joint Conferences on Artificial Intelligence, pp. 3988–3994 (2017)
10. Dutta, S., Das, D.: Dialogue modelling in multi-party social media conversation. In: International Conference on Text, Speech, and Dialogue, pp. 219–227. Springer, Berlin (2017)
11. Eisenlauer, V.: Facebook as a third author—(semi-)automated participation framework in social network sites. J. Pragmat. **72**, 73–85 (2014)
12. Hasan, K.S., Ng, V.: Why are you taking this stance? Identifying and classifying reasons in ideological debates. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 751–762 (2014)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
14. Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., et al.: Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies. IEEE Press, New York (2001)
15. Kalchbrenner, N., Blunsom, P.: Recurrent convolutional neural networks for discourse compositionality. Preprint, arXiv:1306.3584 (2013)
16. Lai, M., Farías, D.I.H., Patti, V., Rosso, P.: Friends and enemies of Clinton and Trump: using context for detecting stance in political tweets. In: Mexican International Conference on Artificial Intelligence, pp. 155–168. Springer, New York (2016)
17. Larson, M.L.: Meaning-Based Translation: A Guide to Cross-Language Equivalence. University press of America, Lanham (1984)
18. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
19. Lotan, G.: Mapping information flows on twitter. In: Proceedings of the ICWSM Workshop on the Future of the Social Web (2011)
20. Malhotra, P., Vig, L., Shroff, G., Agarwal, P.: Long short term memory networks for anomaly detection in time series. In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, pp. 89–94. Presses universitaires de Louvain, Louvain-la-Neuve (2015)
21. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)

22. Misra, A., Walker, M.: Topic independent identification of agreement and disagreement in social media dialogue. Preprint, arXiv:1709.00661 (2017)
23. O'Connor, B., Balasubramanyan, R., Routledge, B.R., Smith, N.A., et al.: From tweets to polls: linking text sentiment to public opinion time series. International Conference on Weblogs and Social Media, vol. 11, no. 122–129, pp. 1–2 (2010)
24. Scott, K.: The pragmatics of hashtags: inference and conversational style on twitter. J. Pragmat. **81**, 8–20 (2015)
25. Somasundaran, S., Namata, G., Wiebe, J., Getoor, L.: Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, vol. 1, pp. 170–179. Association for Computational Linguistics, Morristown (2009)
26. Suh, B., Hong, L., Pirolli, P., Chi, E.H.: Want to be retweeted? Large scale analytics on factors impacting retweet in twitter network. In: 2010 IEEE Second International Conference on Social Computing (socialcom), pp. 177–184. IEEE, Piscataway (2010)
27. Sundermeyer, M., Schlüter, R., Ney, H.: LSTM neural networks for language modeling. In: Thirteenth Annual Conference of the International Speech Communication Association (2012)
28. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)
29. Tan, C., Lee, L., Tang, J., Jiang, L., Zhou, M., Li, P.: User-level sentiment analysis incorporating social networks. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1397–1405. ACM, New York (2011)
30. Trevithick, P., Clippinger, J.H.: Method and system for characterizing relationships in social networks (2008). US Patent 7,366,759
31. Wang, H., Can, D., Kazemzadeh, A., Bar, F., Narayanan, S.: A system for real-time twitter sentiment analysis of 2012 US presidential election cycle. In: Proceedings of the ACL 2012 System Demonstrations, pp. 115–120. Association for Computational Linguistics, Morristown (2012)
32. Wang, K.C., Lai, C.M., Wang, T., Wu, S.F.: Bandwagon effect in Facebook discussion groups. In: Proceedings of the ASE BigData & Social Informatics 2015, p. 17. ACM, New York (2015)
33. Wang, Y., Huang, M., Zhao, L., et al.: Attention-based LSTM for aspect-level sentiment classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 606–615 (2016)
34. Wong, F., Tan, C.W., Sen, S., Chiang, M.: Media, pundits and the US presidential election: quantifying political leanings from tweets. In: Proceedings of the International Conference on Weblogs and Social Media, pp. 640–649 (2013)
35. Zhang, A.X., Culbertson, B., Paritosh, P.: Characterizing online discussion using coarse discourse sequences. In: Proceedings of the Eleventh International Conference on Web and Social Media, pp. 1–10. AAAI Press, Palo Alto (2017)

Chapter 7
Learning from Imbalanced Datasets with Cross-View Cooperation-Based Ensemble Methods

Cécile Capponi and Sokol Koço

Abstract In this paper, we address the problem of learning from imbalanced multi-class datasets in a supervised setting when multiple descriptions of the data—also called views—are available. Each view incorporates various information on the examples, and in particular, depending on the task at hand, each view might be better at recognizing only a subset of the classes. Establishing a sort of cooperation between the views is needed for all the classes to be equally recognized—a crucial problem particularly for imbalanced datasets. The novelty of our work consists in capitalizing on the complementariness of the views so that each class can be processed by the most appropriate view(s), thus improving the per-class performances of the final classifier. The main contribution of this paper are two ensemble learning methods based on recent theoretical works on the use of the confusion matrix's norm as an error measure, while empirical results show the benefits of the proposed approaches.

7.1 Introduction

In machine learning, a frequent issue for datasets coming from real-life applications is the problem of imbalanced classes: some classes (called *majority* classes) are more represented than the others (the *minority* ones). On the other hand, the data may be described by different sets of attributes, also called *views*. The capability of views to deal with a multi-class learning problem is uneven: some views are usually more appropriate than others to process some classes (*cf.* Fig. 7.1) : it is therefore worthwhile to encourage the right views to recognize the right classes, especially in the case of classes that are underpopulated.

The purpose of the work presented in this paper is to address these two imbalanced properties as one: *how to exploit the specificities that each view has*

C. Capponi · S. Koço (✉)
LIS, Aix-Marseille University, Toulon University, CNRS, Marseille, France
e-mail: cecile.capponi@lis-lab.fr

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_7

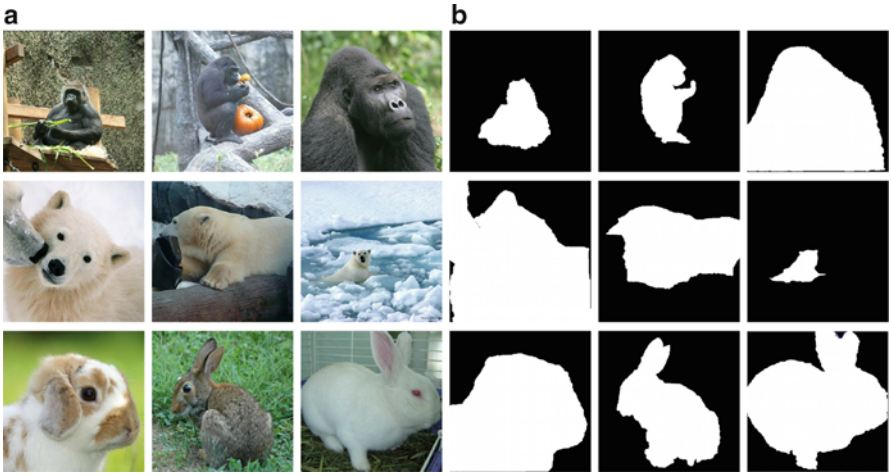


Fig. 7.1 Extract from the dataset *Animals with Attributes* [14]: three examples of classes gorilla, polar bear, and rabbit. One first view reflecting colors (such as histograms of colors) would be sufficient to distinguish polar bears from gorillas, but not enough to discriminate either gorillas from rabbits or rabbits from polar bears. For achieving these two latest purposes, one might expect that image descriptors reflecting edges or segmentation would be better than colors. Multi-class and multi-view classifications are sometimes entangled. (**a**) The original images (the first view is extracted from them, focusing on colors). (**b**) The animals are separated from the background (the second view)

on a per class basis so that the final classifier equally recognizes both majority and minority classes?

To the best of our knowledge, multi-view and imbalanced multi-class supervised classification problems have always been tackled in separate ways. On one side, the supervised multi-view setting is often processed through early fusion of description spaces, or through late fusion of every classifier learnt from the various description spaces [1, 18]. Among the late fusion approaches, the indisputable success of MKL [2] must be balanced with the time required for its processing of high input space dimensions [8]. On the other side, the imbalanced classes problems have been addressed through two main approaches [9, 10]:

- resampling (e.g., SMOTE: [4]) which aims at rebalancing the sample,
- cost-sensitive methods which make use of class-based loss functions for building models that take into account the imbalanced rate of the training set (e.g., [21]).

Both approaches are sometimes coupled with specific feature selection techniques (e.g., [23]) such that the most appropriate features are identified for recognizing the hardest (smallest) classes.

In this paper, we advocate that promoting the cooperation between imbalanced views leads to finding the most appropriate view(s) for each class, thus improving the performances of the final classifier for imbalanced classes problems. Figure 7.2 depicts an example of distributed confusion between classes among views according

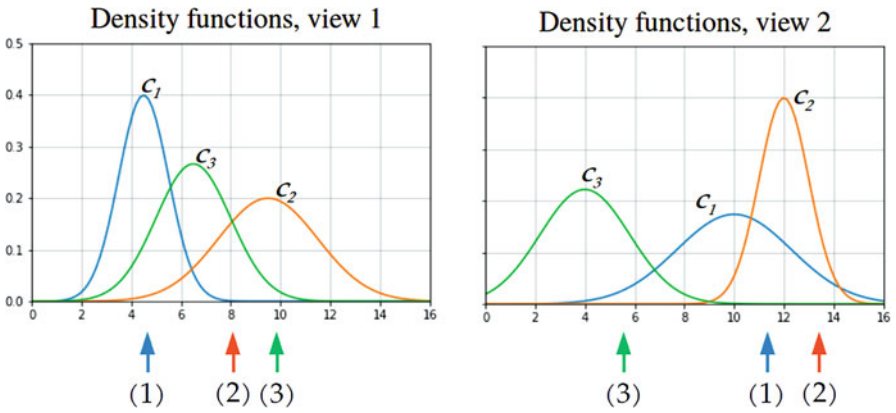


Fig. 7.2 Multi-view multi-class imbalanced confusions in a nutshell. Here are density functions of three classes among two views: the description space is $\mathbb{R}$ in both views ($\mathcal{X} = \mathbb{R} \times \mathbb{R}$), and three examples are pictured according to their description values in each view. Independently from class imbalance (which is not represented), one can easily notice that the confusion rates according to the Bayes error are not balanced within both views: for example, there is more confusion between c_2 and c_3 in view 1 than in view 2. If the Bayes' rule would apply on the first view, then the example 3 would be classified in c_2, whereas on the second view the decision would be c_3. This illustrates that both views should be considered and combined in a proper way in order to decide the right class for each example's data. If we consider the class imbalance problem (which means that $P(y)$ would favor classes over-represented in the dataset), then example 2 is more likely to be classified as c_3 rather than c_2 in view 1 as far as c_2 is under-represented compared to c_3, whereas that confusion is less prominent in view 1

to the Bayes error: such a problem arises in many balanced datasets, and is naturally amplified with class imbalance due to the $P(y)$ of the Bayes' rule.[1] We propose here an algorithm that encourages the cooperation among views/classes in order to select the right view for reducing the confusion between each couple of classes.

Based on recent theoretical results on the use of the confusion matrix's norm as an error measure (e.g., [17]), the aim of the proposed methods is to find the best combination of views for each class ensuring a small confusion norm. Roughly speaking, our proposal is a cost-sensitive method combined with a greedy selection among predefined groups of features (named views).

The remainder of this paper is organized as follows: Sects. 7.2 and 7.3 introduce the notation used in this paper and the motivations and frameworks on which the proposed approaches rely on. The main contribution of this paper is given in Sects. 7.4 and 7.5, where we show step by step how to derive multi-view methods for the imbalanced setting. Finally, Sect. 7.6 gives the experimental results, and we conclude in Sects. 7.7 and 7.8.

[1] The Bayes' rule $B(x)$ chooses y that maximizes $P(x|y)P(y)/P(x)$.

7.2 Theoretical Inspiration from Multi-Class Boosting: Settings

7.2.1 General Notation

Matrices and vectors are denoted by bold capital letters like $\mathbf{C}$ and bold small letters like $\mathbf{x}$, respectively. The entry of the l-th row and the k-th column of $\mathbf{C}$ is denoted $\mathbf{C}(l,k)$, or simply $\mathbf{c}_{l,k}$. $\lambda_{\max}(\mathbf{C})$ and $Tr(\mathbf{C})$, respectively, correspond to the largest eigenvalue and the trace of $\mathbf{C}$. The *spectral* or *operator norm* $\|\mathbf{C}\|$ of $\mathbf{C}$ is defined as:

$$\|\mathbf{C}\| \stackrel{\text{def}}{=} \max_{\mathbf{v}\neq 0} \frac{\|\mathbf{C}\mathbf{v}\|_2}{\|\mathbf{v}\|_2} \stackrel{\text{def}}{=} \sqrt{\lambda_{\max}(\mathbf{C}^*\mathbf{C})},$$

where $\|\cdot\|_2$ is the Euclidian norm and $\mathbf{C}^*$ is the conjugate transpose of $\mathbf{C}$. The inner product and the Frobenius inner product of two matrices $\mathbf{A}$ and $\mathbf{B}$ are denoted $\mathbf{AB}$ and $\mathbf{A}\cdot\mathbf{B}$, respectively.

The indicator function is denoted by $\mathbb{I}$; K is the number of classes, m the number of views, n the number of examples, and n_y the number of examples of class y, where $y \in \{1,\ldots,K\}$. X, Y, and $\mathcal{H}$ are the input, output, and hypothesis spaces, respectively; (x_i, y_i), x_i or i are interchangeably used to denote the ith training example.

7.2.2 Multi-Class Boosting Framework

In this paper we use the boosting framework for multi-class classification introduced in [16], and more precisely the one defined for AdaBoost.MM. Algorithms based on the AdaBoost family maintain a distribution over the training samples in order to identify hard-to-classify examples: the greater the weight of an example, the greater the need to correctly classify these data. In the considered setting, the distribution over the training examples is replaced by a cost matrix. Let $S = \{(x_i, y_i)\}_{i=1}^n$ be a training sample, where $x_i \in X$ and $y_i \in \{1,\ldots,K\}$. The cost matrix $\mathbf{D} \in \mathbb{R}^{n\times K}$ is constructed so that for a given example (x_i, y_i), $\forall k \neq y_i$: $\mathbf{D}(i, y_i) \leq \mathbf{D}(i,k)$, where i is the row of $\mathbf{D}$ corresponding to (x_i, y_i).

This cost matrix is a particular case of cost matrices used in cost-sensitive methods (for example, [20]), where classification costs are given for each example and each class. However, contrary to those methods, the matrix is not given prior to the learning process, but it is updated after each iteration of AdaBoost.MM so that the misclassification cost reflects the difficulty of correctly classifying an example. That is, the costs are increased for examples that are hard to classify, and they are decreased for easier ones.

In AdaBoost.MM, the cost matrix $\mathbf{D}$ at iteration T is defined as follows:

$$\mathbf{D}_T(i,k) \stackrel{\text{def}}{=} \begin{cases} e^{f_T(i,k)-f_T(i,y_i)} & \text{if } k \neq y_i \\ -\sum\limits_{k \neq y_i} e^{f_T(i,k)-f_T(i,y_i)} & \text{otherwise,} \end{cases}$$

where $f_T(i,k)$ is the score function computed as: $f_T(i,k) = \sum_{t=1}^{T} \alpha_t \mathbb{I}[h_t(i) = k]$.

At each iteration t, AdaBoost.MM selects the classifier h and its weight α that minimize the exponential loss:

$$h_t, \alpha_t = \underset{h,\alpha}{\text{argmin}} \sum_{i=1}^{n} \sum_{k \neq y_i} e^{f_t(i,k)-f_t(i,y_i)}, \tag{7.1}$$

$$\text{where: } f_t(i,k) = \sum_{s=1}^{t-1} \alpha_s \mathbb{I}[h_s(i) = k] + \alpha \mathbb{I}[h(i) = k].$$

The final hypothesis of AdaBoost.MM is a weighted majority vote:

$$H(x) = \underset{k=1,\ldots,K}{\text{argmax}} \; f_T(i,k).$$

7.2.3 The Supervised Multi-View Setting

Following previous works [11–13], the multi-view setting that we considered in this paper is supervised. In the case of mono-view classification, where a predictor capable of reliably computing the label associated with some input data, the learning problem hinges on a training set $S = \{(x_i, y_i)\}_{i=1}^{n}$, which is an i.i.d. sample of n observations where $x_i \in \mathcal{X}$, $y_i \in \mathcal{Y}$, and $(x_i, y_i) \sim \mathcal{D}_{\mathcal{X}\times\mathcal{Y}} = \mathcal{D}$. In the multi-view setting, the input space $\mathcal{X}$ is a product of m spaces $\mathcal{X}^{(v)}$, $v = 1, \ldots, m$, and we note $\mathcal{D}^{(v)} \sim P(X^{(v)}, Y)$ where $X^{(v)}$ is a random variable that ranges in $\mathcal{X}^{(v)}$.

Multi-view learning is the usual problem of learning $\hat{h} \in \mathcal{H} : \mathcal{X} \to \mathcal{Y}$ such that the generalization risk is minimized, i.e.,

$$\hat{h} = \text{argmin}_{h\in\mathcal{H}} \, \mathcal{R}(h) = \text{argmin}_{h\in\mathcal{H}} \, \mathbb{E}_{\sim\mathcal{D}} \left[\ell(h(x), y)\right]$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ is some loss function, for example, the 0–1 loss: $\ell(y, y') = \mathbf{1}_{y\neq y'}$.

For each view v, let $\hat{h}^{(v)} : \mathcal{X}^{(v)} \to \mathcal{Y}$ be such that

$$\hat{h}^{(v)} = \text{argmin}_{h\in H^{(v)}} \, \mathbb{E}_{\sim\mathcal{D}^{(v)}} \left[\ell(h^{(v)}(x), y)\right]$$

In order for the multi-view learning to be worthwhile, we obviously expect that $\forall v \in [1, \ldots, m]$,

$$\mathbb{E}_{\sim \mathcal{D}^{(v)}} \left[\ell(\hat{h}^{(v)}(x), y) \right] > \mathbb{E}_{\sim \mathcal{D}} \left[\ell(\hat{h}(x), y) \right]$$

which means that it is beneficial to consider several views than the best one.

7.3 Reducing the Class Confusion with Multi-View Insights

7.3.1 Confusion Matrix: A Probabilistic Definition

When dealing with imbalanced classes, a common tool used to measure the goodness of a classifier is the confusion matrix. Previous works on the confusion matrix (e.g., [15, 17], etc.), advocate that using the operator norm of (a particular formulation of) the confusion matrix as an optimization criterion is a reasonable choice for dealing with the imbalanced classes problem. Following in their steps, in this paper we consider a particular definition of the confusion matrix:

Definition 1 For a given classifier $H \in \mathcal{H}$, an unknown distribution $\mathcal{D}$ over $X \times Y$, and a training set $S = \{(x_i, y_i)\}_{i=1}^{n}$ *i.i.d* according to $\mathcal{D}$, the true and empirical confusion matrices of h, denoted $\mathbf{C} = (\mathbf{c}_{l,k})_{1 \leq l,k \leq K}$ and $\mathbf{C}_S = (\hat{\mathbf{c}}_{l,k})_{1 \leq l,k \leq K}$, respectively, are defined as:

$$\mathbf{c}_{l,k} \overset{\text{def}}{=} \begin{cases} 0 & \text{if } l = k \\ \mathbb{P}_{(x,y)\sim\mathcal{D}}(H(x) = k | y = l) & \text{otherwise.} \end{cases}$$

$$\hat{\mathbf{c}}_{l,k} \overset{\text{def}}{=} \begin{cases} 0 & \text{if } l = k \\ \displaystyle\sum_{i=1}^{n} \frac{1}{n_l} \mathbb{I}[H(x_i) = k] \mathbb{I}[y_i = l] & \text{otherwise,} \end{cases}$$

Contrary to the usual (probabilistic) definition of the confusion matrix, the diagonal entries are zeroed. The advantage of this formulation is twofold: first, it takes into account only the errors of the classifier, and second, its operator norm gives a bound on the true risk of the classifier. Indeed, let $\mathbf{p} = [P(y = 1), \ldots, P(y = K)]$ be the vector of class priors distribution, then we have:

$$R(h) \overset{\text{def}}{=} P_{(x,y)\sim\mathcal{D}}(H(x) \neq y) = \|\mathbf{pC}\|_1 \leq \sqrt{K} \|\mathbf{C}\|, \tag{7.2}$$

where $R(h)$ is the true risk of h and $\| \cdot \|_1$ denotes the $l1$-norm. The aim of the methods presented is this paper is thus to find a classifier $\hat{H}$ that verifies the following criterion:

$$\hat{H} = \underset{H \in \mathcal{H}}{\operatorname{argmin}} \|\mathbf{C}\|. \tag{7.3}$$

7.3.2 From the Confusion Matrix Norm to an Optimization Problem

The confusion matrix $\mathbf{C}$, as given in Definition 1, depends on the unknown distribution $\mathcal{D}$, thus making the problem given in Eq. (7.3) difficult to tackle directly. In order to bypass this, we make use of Theorem 1 in [17], which bounds the norm of the *true* confusion matrix by the norm of its empirical estimation. We give here a reformulation of the theorem for the supervised setting, where the considered loss is the indicator function $\mathbb{I}$.

Corollary 1 *For any $\delta \in (0; 1]$, it holds with probability $1 - \delta$ over a sample $S_{(\mathbf{x},y)\sim\mathcal{D}}$ that:*

$$\|\mathbf{C}\| \leq \|\mathbf{C}_S\| + \sqrt{2K \sum_{k=1}^{K} \frac{1}{n_k} \log \frac{K}{\delta}},$$

where $\mathbf{C}_S$ is the empirical confusion matrix computed for a classifier h over S.

The direct implication of Corollary 1 is that minimizing the norm of the empirical confusion matrix results in the minimization of its true norm. Unfortunately, due to the nature of the confusion matrix in Definition 1, an analytical expression for the norm of the matrix is difficult to compute. We thus consider an upper bound on $\|\mathbf{C}_S\|^2$:

$$\|\mathbf{C}_S\|^2 = \lambda_{\max}(\mathbf{C}_S^* \mathbf{C}_S) \leq Tr(\mathbf{C}_S^* \mathbf{C}_S) \leq \sum_{l=1}^{K} \sum_{k \neq l} \hat{\mathbf{c}}_{l,k} = \|\mathbf{C}_S\|_1. \tag{7.4}$$

In the last part of Eq. (7.4), we abuse the notation and denote the entry-wise $l1$-norm of the matrix by $\|\mathbf{C}\|_1$. Equation (7.4) implies that the updated goal is:

$$\hat{H} = \underset{H \in \mathcal{H}}{\operatorname{argmin}} \|\mathbf{C}\|_1. \tag{7.5}$$

Another drawback of the confusion matrix as given in Definition 1 is the presence of the indicator function, which is not optimization friendly. One way to handle this is to replace the indicator function with loss functions $\ell_{l,k}(H, x)$ defined over two classes, so that $\forall (x, y) \in S$ and $l, k \in \{1, \ldots, K\}$, $\mathbb{I}(h(x) \neq y) \leq \ell_{l,k}(H, x)$. Applying these losses to Eq. (7.4), we have the actual upper bound of the confusion matrix's norm:

$$\|\mathbf{C}\|_1 \leq \sum_{i=1}^{n} \sum_{k \neq y_i} \frac{1}{n_{y_i}} \ell_{y_i,k}(H, \mathbf{x}_i). \tag{7.6}$$

7.3.3 A Multi-View Classifier

In the multi-view setting, an example is represented by various sets of attributes, called views. Although they represent the same objects, different views might be suited for different tasks and/or categories (classes) of objects. For instance, in image classification, the view *color* is more suited for distinguishing between gorillas and polar bears, than between cats and dogs. The motivation behind the work in this paper is to deal with both *imbalanced data* (some classes are more represented than others in the training set) and *imbalanced views* (some views are suited only for a subset of the classes). While the optimization problem given in Eq. (7.5) deals with the imbalanced nature of the dataset, the multi-view aspect has not yet been considered.

Assuming that each view is more adapted only for a subset of classes, one possible way to implement this in a classifier is to associate a coefficient to each view based on its prediction. The better a view v recognizes a class c, the higher this coefficient $\beta_c^{(v)}$ should be. More precisely, the considered classifier is the following:

$$H(x) = \underset{c\in 1,\ldots,K}{\operatorname{argmax}} \sum_{v=1}^{m} \beta_c^{(v)} \mathbb{I}[h^{(v)}(x) = c] \tag{7.7}$$

$$\text{where } \forall c \in \{1, \ldots, K\}, \sum_{v=1}^{m} \beta_c^{(v)} = 1 \text{ and } \beta_c^{(v)} \in [0, 1]. \tag{7.8}$$

In Eq. (7.7), $h^{(v)}$ denotes the classifier learnt on view v and $\beta_c^{(v)}$ can be seen as the confidence that view v gets right for class c (hence the need for these coefficients to be positives). Due to the sum condition imposed in Eq. (7.8), the coefficients $\beta_c^{(v)}$ also define how the views cooperate one with another, and we refer to them as *cooperation coefficients*.

In the following sections, we describe step by step how to learn a cooperation-based multi-view classifier as defined in Eq. (7.7), which is a solution to the optimization problem in Eq. (7.5) and subjected to the conditions of Eq. (7.8).

7.4 Multi-View Classification with Cooperation

The multi-view classifier in Eq. (7.7) associates to an example i the class k that obtains the highest score $\phi(i, k)$, computed as follows:

$$\phi(i, k) = \sum_{v\in\{1,\ldots,m\}} \beta_k^{(v)} \mathbb{I}[h^{(v)}(i) = k]$$

If an example i is misclassified, then there exists at least one class $k \neq y_i$ such that $\phi(i,k) \geq \phi(i,y_i)$, that is, $e^{\phi(i,k)-\phi(i,y_i)} \geq 1$. On the other hand, i correctly classified implies that $\forall k \neq y_i, \phi(i,k) \leq \phi(i,y_i)$ and $0 < e^{\phi(i,k)-\phi(i,y_i)} \leq 1$. Basically, the exponential loss based on the score functions ϕ satisfies the conditions put on the losses in Eq. (7.6). Injecting these exponential losses in Eq. (7.6), we have:

$$\begin{aligned}\|\mathbf{C}_{\mathcal{S}}\|_1 &\leq \sum_{i=1}^{n}\sum_{k\neq y_i}\frac{1}{n_{y_i}}\ell_{y_i,k}(h,x_i) \leq \sum_{i=1}^{n}\sum_{k\neq y_i}\frac{1}{n_{y_i}}e^{\phi(i,k)-\phi(i,y_i)} \\ &= \sum_{i=1}^{n}\sum_{k\neq y_i}\frac{1}{n_{y_i}}e^{\sum_{v=1}^{m}(\beta_k^{(v)}\mathbb{I}[h^{(v)}(i)=k]-\beta_{y_i}^{(v)}\mathbb{I}[h^{(v)}(i)=y_i])}\end{aligned} \tag{7.9}$$

A first approach to minimizing the norm of the confusion matrix is to find the classifiers and the cooperation coefficients that are the solution of the optimization problem given in Eq. (7.9). This formulation is quite similar to the most general formulation of multiple kernel learning (MKL) methods [2], where the kernels and the weighting coefficients are learnt at the same time. The differences are that in our case, the classifiers are not limited to kernels, the number of coefficients is higher, and we use an exponential loss function.

The main advantage of this approach is that the cooperation between the views is promoted both in the training phase of the classifiers and in the choice of the coefficients. However, on the downside, the classifiers need to be learnt at the same time—which can make the learning procedure quite tedious—and it is not clear what the minimization goal for each classifier is. A friendlier (and easier to interpret) expression for Eq. (7.9) can be obtained by limiting the cooperation between the views only to the choice of the coefficients. The following result is a key step in this direction.

Lemma 1 *Let $n \in \mathbb{N}$ and $a_1, \ldots, a_n \leq 1$, so that $\sum_{j=1}^{n} a_j \leq 1$. Then the following inequality is true:*

$$\exp\left(\sum_{j=1}^{n} a_j\right) \leq \sum_{j=1}^{n}\exp(a_j). \tag{7.10}$$

Proof The proof for this result is realized by induction : first we show that the result holds for two variables, then we prove the general case.

The first condition we set on the coefficients in the optimization problem of Eq. (7.8) is that their values should be in [0, 1]. The second condition requires that, for each class, the coefficients sum up to 1. Together, these conditions imply that the inner sums in Eq. (7.9) take their values in $[-1, 1]$, same as each of their elements. We can thus apply Lemma 1 to further simplify the expression given in Eq. (7.9).

$$\begin{aligned}\|\mathbf{C}_{\mathcal{S}}\|_1 &\leq \sum_{i=1}^{n}\sum_{k\neq y_i}\frac{1}{n_{y_i}}e^{\sum_{v=1}^{m}(\beta_k^{(v)}\mathbb{I}[h_j(i)=k]-\beta_{y_i}^{(v)}\mathbb{I}[h^{(v)}(i)=y_i])} \\ &\leq \sum_{i=1}^{n}\sum_{k\neq y_i}\frac{1}{n_{y_i}}\sum_{v=1}^{m}e^{\beta_k^{(v)}\mathbb{I}[h^{(v)}(i)=k]-\beta_{y_i}^{(v)}\mathbb{I}[h^{(v)}(i)=y_i]} \\ &= \sum_{v=1}^{m}\sum_{i=1}^{n}\sum_{k\neq y_i}\frac{e^{\beta_k^{(v)}\mathbb{I}[h^{(v)}(i)=k]-\beta_{y_i}^{(v)}\mathbb{I}[h^{(v)}(i)=y_i]}}{n_{y_i}} \leq \sum_{v=1}^{m}\ell(h^{(v)}),\end{aligned} \tag{7.11}$$

where $\ell(h^{(v)}) = \sum_{i=1}^{n}\sum_{k\neq y_i}\frac{1}{n_{y_i}}e^{\beta_k^{(v)}\mathbb{I}[h^{(v)}(i)=k]-\beta_{y_i}^{(v)}\mathbb{I}[h^{(v)}(i)=y_i]}$ defines the loss of the classifier $h^{(v)}$—whose predictions are weighted by the coefficients $\beta_k^{(v)}, \forall k \in \{1,\ldots,K\}$—on the training set S.

Minimizing the norm of the confusion matrix for a multi-view classifier H, as defined in Eq. (7.7), ends up finding the classifiers and coefficients that minimize the loss $\sum_{v=1}^{m}\ell(h^{(v)})$. Remark that for fixed values of the coefficients, Eq. (7.11) suggests that for each view, it suffices to find the classifier minimizing the loss $\ell(\cdot)$, instead of finding the classifiers that minimize a loss depending on their combination (which was the case in Eq. (7.9)).

Although the training procedure where all the coefficients and classifiers are learnt at the same time is still a viable solution, the previous remark suggests that a two-step procedure is better adapted for this case. The first step consists in finding, for each view, the classifier whose error $\ell(\cdot)$ is minimal. The second step consists in finding the coefficients that minimize the whole loss (Eq. (7.11)). Due to the formulation of Eq. (7.11), the coefficients of one class can be computed independently from the coefficients of the other classes. More precisely, let $S_+^{(v)}$ denote the set of examples correctly classified by $h^{(v)}$ and $S_-^{(v)}$ the set of misclassified ones. The right-handed side of Eq. (7.11) can be written as:

$$\begin{aligned}&\sum_{v=1}^{m}\sum_{i=1}^{n}\sum_{k\neq y_i}\frac{1}{n_{y_i}}e^{\beta_k^{(v)}\mathbb{I}[h^{(v)}(i)=k]-\beta_{y_i}^{(v)}\mathbb{I}[h^{(v)}(i)=y_i]} \\ &= \sum_{v=1}^{m}\sum_{i\in S_+^{(v)}}\frac{K-1}{n_{y_i}}e^{-\beta_{y_i}^{(v)}} + \sum_{v=1}^{m}\sum_{i\in S_-^{(v)}}\left(\frac{K-2}{n_{y_i}} + \frac{e^{-\beta_{h^{(v)}(i)}^{(v)}}}{n_{y_i}}\right) \\ &= \sum_{c=1}^{K}\sum_{v=1}^{m}\left(e^{-\beta_c^{(v)}}A + e^{\beta_c^{(v)}}B\right) + C,\end{aligned} \tag{7.12}$$

where:

$$A = \sum_{i \in S_+^{(v)}} \frac{K-1}{n_{y_i}} \mathbb{I}[y_i = c], \; B = \sum_{i \in S_-^{(v)}} \frac{1}{n_{y_i}} \mathbb{I}[h^{(v)}(i) = c]$$

$$\text{and } C = \sum_{c=1}^{K} \sum_{v=1}^{m} \sum_{i \in S_-^{(v)}} \frac{K-2}{n_{y_i}} \mathbb{I}[h^{(v)}(i) = c].$$

The first equality is obtained by splitting the training sample in $S_+^{(v)}$ and $S_-^{(v)}$, for all views v. In the third equality, we replace y_i and $h^{(v)}(i)$ by a label c, which allows us to regroup the non-constant terms (that is, those depending on $\beta_c^{(v)}$).

Equation (7.12) suggests that for a given class c, the coefficients $\beta_c^{(v)}, \forall v \in \{1, \ldots, m\}$ should be the ones that minimize the *per class* loss $\ell(c)$:

$$\sum_{v=1}^{m} \left(e^{-\beta_c^{(v)}} A + e^{\beta_c^{(v)}} B \right). \tag{7.13}$$

The pseudo-code of the two-step method is given in Algorithm 1. The *stopping criterion* can be related to the empirical loss of the classifier computed on S, or the drop of the loss in Eq. (7.12), and so on. Contrary to the MKL-like approach suggested in Eq. (7.9), in the two-step method, the classifiers can be learnt in parallel, reducing the training time for the algorithm.

7.5 Boosting The Cooperation

7.5.1 *A Multi-View Boosting Method from the Confusion Matrix Norm Minimization*

The two-step procedure in Algorithm 1 suggests that at each iteration the learnt classifiers should minimize some training error weighted by cooperation coefficients associated with the training examples. More precisely, the training procedure for each view consists in learning a classifier that minimizes a weighted empirical error, re-weighting the examples based on the performances of the classifier and reiterating until a stopping criterion is met. Interestingly, this procedure is fairly similar to iterative boosting methods, such as AdaBoost [6] and its multi-class formulation AdaBoost.MM, recalled in Sect. 7.2.2. In this section, we study the case where the classifiers $h^{(v)}$ in Eq. (7.11) are replaced with boosted classifiers.

Algorithm 1 Carako

Given :
S = $\{(x_i, y_i)\}_{i=1}^{n}$ where $x_i \in \mathcal{X}^{(1)} \times \mathcal{X}^{(2)} \times \ldots \times \mathcal{X}^{(m)}$, $y_i \in \{1, \ldots, K\}$
Initialise :
For all $1 \leq c \leq K$, $1 \leq v \leq m$, $\beta_c^{(v)} =$ rand(0,1) (w.r.t. to Eq. (7.8))

while *stopping criterion* not met:

1. $\forall v \in \{1, \ldots, m\}$, train $h^{(v)}$ minimizing the loss $\ell(h^{(v)})$ (Eq. (7.11))
2. $\forall c \in \{1, \ldots, K\}$, compute $\beta_c^{(v)}$ minimizing the loss $\ell(c)$ (Eq. (7.13)), *w.r.t.* conditions in Eq. (7.8)

Output :

$$H(\cdot) = \text{argmax}_{c \in \{1,\ldots,K\}} \sum_{v=1}^{m} \beta_c^{(v)} \mathbb{I}[h^{(v)}(\cdot) = c]$$

An iterative boosting method runs for T rounds and its output hypothesis is computed as follows:

$$h(\cdot) = \text{argmax}_{c \in \{1,\ldots,K\}} \sum_{t=1}^{T} \alpha_t \mathbb{I}[h_t(\cdot) = c],$$

where h_t are classifiers performing slightly better than random guessing (also called *weak classifiers*) and α_t are positive real-valued coefficients that represent the importance given to h_t. The main advantage of weak classifiers is that they can be used to exploit localized informations. Thus the motivation for replacing the per-view classifiers in Eq. (7.11) with multiple weak classifiers learnt on the views is to better use the localized information in each view, in particular information related to how the view recognizes the various classes.

When defining the multi-view classifier in Eq. (7.7), we argued that each classifier should be associated with an importance coefficient depending on the prediction. However, having a single coefficient per class and view might not be a good strategy when dealing with boosted classifiers, since each of the weak classifiers has different performances. Thus we propose to associate to each classifier not only its importance coefficient, but also coefficients depending on its actual prediction:

$$h^{(v)}(\cdot) = \text{argmax}_{c \in \{1,\ldots,K\}} \sum_{t=1}^{T} \alpha_t^{(v)} \beta_{t,c}^{(v)} \mathbb{I}[h_t^{(v)}(\cdot) = c], \text{ where } \beta_{t,c}^{(v)} \in [0, 1]. \tag{7.14}$$

In this case the coefficient associated with a view v for a class c is:

$$\beta_c^{(v)} = \frac{\sum_{t=1}^{T} \alpha_t^{(v)} \beta_{t,c}^{(v)}}{\sum_{t=1}^{T} \alpha_t^{(v)}}.$$

The condition $\forall t \in \{1,\ldots,T\}, \forall c \in \{1,\ldots,K\}, \sum_{v=1}^{m} \beta_{t,c}^{(v)} = 1$ ensures that $\beta_c^{(v)}$ is always smaller than 1. The downside is that it does not guarantee that $\forall c \in \{1,\ldots,K\}, \sum_{v=1}^{m} \beta_c^{(v)} = 1$, but rather that $\sum_{v=1}^{m} \beta_c^{(v)} \leq 1$.

The classifier defined in Eq. (7.14) is similar to the classifier defined in Eq. (7.7): for a given example, it computes a score for each class. In particular for examples in the training sample $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, these scores correspond to:

$$f_T^{(v)}(i,c) = \sum_{t=1}^{T} \alpha_t^{(v)} \beta_{t,c}^{(v)} \mathbb{I}[h_t^{(v)}(i) = c], \text{ for } 1 \leq v \leq m, 1 \leq c \leq K, 1 \leq i \leq n.$$

Armed with these score functions, we are now ready to tackle the last optimization problem in this paper. Continuing from Eq. (7.11), we have:

$$\begin{aligned}
\|\mathbf{C}_{\mathcal{S}}\|_1 &\leq \sum_{v=1}^{m}\sum_{i=1}^{n}\sum_{c\neq y_i} \frac{1}{n_{y_i}} e^{\beta_c^{(v)}\mathbb{I}[h^{(v)}(i)=c] - \beta_{y_i}^{(v)}\mathbb{I}[h^{(v)}(i)=y_i]} \\
&\leq \sum_{v=1}^{m}\sum_{i=1}^{n}\sum_{c\neq y_i} \frac{1}{n_{y_i}} e^{\mathbb{I}[h^{(v)}(i)=c]} \leq \sum_{v=1}^{m}\sum_{i=1}^{n}\sum_{c\neq y_i} \frac{1}{n_{y_i}} e^{\ln(2+\exp[f_T^{(v)}(i,c) - f_T^{(v)}(i,y_i)])} \\
&= \sum_{v,i,c\neq y_i} \frac{1}{n_{y_i}} e^{\Delta_{f,T}(v,c,y_i)} + 2mK(K-1), \qquad (7.15)
\end{aligned}$$

where $\Delta_{f,T}(v,c,y_i) = f_T^{(v)}(i,c) - f_T^{(v)}(i,y_i)$.

The second inequality follows from the fact that the coefficients $\beta_c^{(v)}$ are positive and at most 1. For the third inequality a particular case of the logistic loss is used. If $h^{(v)}$ predicts class c, for example, i ($\mathbb{I}[h^{(v)}(i) = c] = 1$), then $f_T^{(v)}(i,c) \geq f_T^{(v)}(i,y_i)$ and $\exp(f_T^{(v)}(i,c) - f_T^{(v)}(i,y_i)) \geq 1$. Since the difference between the scores may be infinitely small, we use 2 in the logistic loss, which ensures $\ln(2 + \exp[f_T^{(v)}(i,c) - f_T^{(v)}(i,y_i)]) \geq 1$. In the case where $h^{(v)}$ predicts another class for i, other than c, then $\ln(2 + \exp[f_T^{(v)}(i,c) - f_T^{(v)}(i,y_i)]) > 0$.

We have thus:

$$\|\mathbf{C}_{\mathcal{S}}\|_1 \leq \sum_{v=1}^{m} \ell(h^{(v)}) + 2mK(K-1), \qquad (7.16)$$

where $\ell(h^{(v)}) = \sum_{i=1}^{n}\sum_{c\neq y_i} \frac{1}{n_{y_i}} \exp(f_T^{(v)}(i,c) - f_T^{(v)}(i,y_i))$ defines the loss of the combinations of all the classifiers learnt on view v. Equation (7.16) suggests that at each iteration, for each view, a classifier that minimizes the loss $\ell(h_j)$ should be learnt. It is interesting to notice that the loss $\ell(h^{(v)})$ is quite similar to the loss of AdaBoost.MM, as given in Eq. (7.1) in Sect. 7.2, except for the coefficients $\beta_{c,t}^{(v)}$

and the re-weighting coefficients $\frac{1}{n_{y_i}}$. In other words, an AdaBoost.MM-like process should take place in each view. It follows that a cost matrix should be maintained for each view and the classifier (and its importance coefficient) should be computed in a similar way as in AdaBoost.MM or in MuMBo [13].

Similarly to the two-step procedures in Carako (Algorithm 1), the algorithm μCoMBo derived from the minimization of the loss in Eq. (7.16) uses a two-step procedure at each iteration: first a classifier is learnt for each view and the importance coefficient is computed as in Algorithm 2, and second, the cooperation coefficients are computed so that they minimize the loss. The pseudo-code of μCoMBo is given in Algorithm 2. Due to the boosting nature of the method, the stopping criterion used is the number of iterations.

Algorithm 2 μCoMBo: MUlti-view COnfusion Matrix BOosting

Given

- $\mathrm{S} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ where $x_i \in \mathcal{X}^{(1)} \times \ldots \times \mathcal{X}^{(m)}$, $y_i \in \{1, \ldots, K\}$
- T the number of iterations
- $\forall i \in \{1, \ldots, n\}, \forall v \in \{1, \ldots, m\}, \forall c \in \{1, \ldots, K\}$

$$f_0^{(v)}(i, c) = 0, \beta_{0,c}^{(v)} = 1/m \text{ and}$$

$$\mathbf{D}_0^{(v)}(i, c) = \begin{cases} \frac{1}{n_{y_i}} & \text{if } c \neq y_i \\ -\frac{K-1}{n_{y_i}} & \text{if } c = y_i \end{cases}$$

for $t = 1$ **to** T **do**

$\forall v$: Get $h_t^{(v)}$ and $\alpha_t^{(v)} = \frac{1}{2} \ln \frac{1+\delta_t^{(v)}}{1-\delta_t^{(v)}}$, where $\delta_t = \frac{-\sum_{i=1}^n \mathbf{D}_{t-1}^{(v)}(i, h_t^{(v)}(\mathbf{x}_i))}{\sum_{i=1}^n \sum_{c \neq y_i} \mathbf{D}_{t-1}^{(v)}(i,c)}$

Compute $\beta_{t,c}^{(v)}, \forall v \in \{1, \ldots, m\}, c \in \{1, \ldots, K\}$ minimizing Eq. (7.15)

Update cost matrices (for each $v = 1, \ldots, m$):

$$\mathbf{D}_t^{(v)}(i, c) = \begin{cases} \frac{1}{n_{y_i}} e^{f_t^{(v)}(i,c) - f_t^{(v)}(i,y_i)} & \text{if } c \neq y_i \\ -\sum_{l \neq y_i}^{K} \frac{e^{f_t^{(v)}(i,l) - f_t^{(v)}(i,y_i)}}{n_{y_i}} & \text{if } c = y_i \end{cases}$$

where $f_t^{(v)}(i, c) = \sum_{z=1}^{t} \alpha_z^{(v)} \beta_{z,c}^{(v)} \mathbb{I}1[h_z^{(v)}(i) = c]$

end for

Output final hypothesis :

$$H(\cdot) = \text{argmax}_{c \in \{1, \ldots, K\}} \sum_{t=1}^{T} \sum_{v=1}^{m} \beta_{t,c}^{(v)} \alpha_t^{(v)} \mathbb{I}[h_t^{(v)}(\cdot) = c]$$

7.5.2 On the Theoretical Properties of μCoMBo

One of the most important properties of (adaptive) boosting algorithms is that the total loss (Eq. (7.1) for AdaBoost.MM) drops at each iteration, provided that the chosen classifier weight verifies the boostability conditions. In the binary case, the classifier is required to perform slightly better than random guessing (that is, its error should be smaller than 0.5). For multi-class algorithms, such as AdaBoost.MM and μCoMBo, the condition consists in finding classifiers whose classification cost (that is, error) should be smaller that the one for an arbitrary baseline. Theorem 1 states that, for a given view, if the selected classifier verifies the multi-class boostability condition, then the total loss of μCoMBo (computed from that view) decreases.

Theorem 1 *For a given view $v \in \{1, \ldots, m\}$, suppose the cost matrix $\mathbf{D}_t^{(v)}$ is chosen as in the Algorithm 2, and the returned classifier $h_t^{(v)}$ satisfies the edge condition for the baseline $\mathbf{U}_{\delta_t^{(v)}}$ and cost matrix $\mathbf{D}_t^{(v)}$,* i.e., *$\mathbf{D}_t^{(v)} \cdot \mathbf{1}_{h_t^{(v)}} \leq \mathbf{D}_t^{(v)} \cdot \mathbf{U}_{\delta_t^{(v)}}$, where $\mathbf{1}_h$ is the matrix defined as $\mathbf{1}_h(i, l) = \mathbb{I}[h(i) = l]$.*

Then choosing a weight $\alpha_t^{(v)} > 0$ for $h_t^{(v)}$ allows

$$\ell_t(h^{(v)}) \leq \kappa_t^{(v)} \ell_{t-1}(h^{(v)}),$$

to hold, with:

$$\kappa_t^{(v)} = 1 - \frac{1}{2}\left(e^{\alpha_t^{(v)}} - e^{-\alpha_t^{(v)}}\right)\delta_t^{(v)} + \frac{1}{2}\left(e^{\alpha_t^{(v)}} + e^{-\alpha_t^{(v)}} - 2\right)$$

Proof The proof is similar to the one provided for AdaBoost.MM in [16].

The result given in Theorem 1 implies that choosing the importance coefficient as in Algorithm 2, the drop of the loss for a view v is $\sqrt{1 - \delta_t^{(v)}}$. That is, at each iteration, after the first step which consists in choosing a classifier per view, the total drop in loss is:

$$\sum_{v=1}^{m} \ell_t(h^{(v)}) \leq \sum_{v=1}^{m} \sqrt{1 - \delta_t^{(v)}}\, \ell_{t-1}(h^{(v)}). \tag{7.17}$$

As long as the classifiers learnt on the views achieve positive edges on their corresponding cost matrices, the whole loss is guaranteed to decrease.

Note that in the right side of Eq. (7.17), the loss $\ell_{t-1}(h^{(v)})$ depends on the score functions defined as:

$$f_t^{(v)}(i, c) = \sum_{z=1}^{t-1} \alpha_z^{(v)} \beta_{z,c}^{(v)} \mathbb{I}[h_z^{(v)}(i) = c] + \alpha_t^{(v)} \beta_{t,c}^{(v)} \mathbb{I}[h_t^{(v)}(i) = c],$$

since the classifiers are learnt before the cooperation coefficients β, that is, we simply suppose that all these coefficients are equal to 1. Obviously the drop in the final loss, after the second step of iteration t, is much bigger given that all the β take their values in $[0, 1]$ and at most one of the $\beta_{t,c}^{(v)}$ is at 1 for a given class c. However, since these coefficients are computed as a solution to an optimization problem, finding an analytical expression for the actual drop after iteration t is quite challenging.

7.6 Experimental Results

Experimental results of algorithms CARAKO and μCoMBo are compared with results obtained with state-of-the-art methods.

These experiments are intended to ascertain the relevance of the herein proposed algorithms, by checking the experimental gaps achieved when comparing mono and multi-view confusion imbalanced within a multi-class setting. As such, no tuning of hyperparameters was done (default hyperparameters are considered), and only one—reduced—dataset was considered.

In order to have a first insight into the relevance of the approaches, we processed experiments on a subset of the Animal with Attributes (AwA) images open database proposed in [14], because it regroups the two problems addressed in this paper: (1) AwA is highly imbalanced: some classes are way more represented than others, and (2) AwA comes with six pre-extracted feature representations (thus, views) for each image, related to different properties of the images.

7.6.1 AwA Presentation and Experimental Protocols

Originally, AwA comes with 50 classes and six views on images. We extracted[2] six classes, from the less represented to the most populated; class names (and number of examples) are: *beaver* (184), *buffalo* (559), *deer* (1072), *gorilla* (802), *lion* (483), and *polar+bear* (815). Four views (for a total of 6940 real attributes) among six were selected both on the nature of information contained therein (local versus global) and the possibility of the view to recognize all the classes or some of them. More precisely, we consider: color histogram features (2688 attributes), local self-similarity features (2000 attributes), PyramidHOG (PHOG) (252 attributes), and scale-invariant feature transform features (2000 attributes).

Some examples of the animal classes are given in Fig. 7.3.

[2]The original dataset comes with a high number of examples (more than 30,000), classes (50), and real features (10,520). Currently, intensive experiments are on their way, with grid-search hyperparameters tuning over more learning methods, running over the while AwA datasets: it should take several months.

Fig. 7.3 Some examples of the six selected classes of AwA

Classifiers Six methods are tested: (1–2) early fusion: AdaBoost.MM and μCoMBo on the concatenation of the views, (3–4) late fusion: AdaBoost.MM and CARAKO, (5) multi-view methods: μCoMBo, and (6) multiclassMKL from the Shogun toolbox [19].

For the boosting-based methods, we used 1-level decision trees (stumps) as weak learners and they were run for 100 iterations; for Carako, the depth of the trees was limited to 10. The MKL were learnt with gaussian kernels on each view (same parameters: mean=8, width=2), with a regularized L1 norm and the regularization parameter $C = 1.0$, and $\epsilon = 0.001$.

Evaluation Protocol Through a fivefold cross-validation process, we evaluated each classifier along five measures: per-class recall, overall accuracy, MAUC, G-mean, and norm of the confusion matrix. If K is the number of classes:

$$\text{MAUC} = \sum_{i \neq j} \frac{\text{AUC}_{i,j}}{K(K-1)} \text{ and G-mean} = \left(\prod_{j=1,\ldots,K} \text{recall}_j \right)^{\frac{1}{K}}$$

G-mean and MAUC consider each class independently from its population in the learning sample; a G-mean is zero whenever one minor class has a zero recall.

7.6.2 Performance Results

Table 7.1 gives the results for accuracy, G-mean, MAUC, and norm of the confusion matrix, as well as an indication of the training time.[3] About the overall accuracy,

[3]The multiclassMKL is quite long for it is a QCQP problem, hardly depending on the number of classes and views (kernels); boosting approaches benefits from the possibility of parallelizing the weak classifiers training.

Table 7.1 Overall accuracy, per-class recall, MAUC, G-mean, and norm of the confusion matrix for the six methods on AwA (the suffix `-e` indicates an early fusion)

		beaver	*buffalo*	*deer*	*gorilla*	*lion*	*p. bear*				
Algos	Acc.	4.7%	14.3%	27.4%	20.5%	12.3%	20.8%	MAUC	G-mean	$\|\mathbf{C}\|$	Time
Ada.MM-e	40.2%	0.0%	0.0%	77.8%	68.1%	0.0%	22.8%	0.694	0.000	1.385	90
Ada.MM-l	44.8%	0.0%	0.0%	**93.7%**	34.9%	0.0%	57.9%	0.731	0.000	1.594	71
Carako	34.4%	23.1%	20.1%	36.4%	33.5%	25.0%	50.7%	0.631	0.069	0.964	122
MKL	**58.6%**	0.0%	19.9%	77.3%	73.4%	13.6%	**85.6%**	0.669	0.000	0.980	2825
μCoMBo-e	43.4%	28.6%	30.8%	28.3%	66.9%	38.7%	55.6%	0.731	0.365	0.899	98
μCoMBo	55.5%	**44.5%**	**35.3%**	48.3%	**71.9** %	**38.8** %	74.4%	**0.821**	**0.481**	**0.552**	75

Results in boldface indicate when one algorithm is significantly better than the others (student test with 95% confidence). The rate of train examples is given under each class. The training time of a 10-fold cross-validation is given in minutes, on a 4-core processor with limited RAM

the better performing method is multiclassMKL, while the recalls per class indicate that MKL focus on majority class, mechanically improving the overall accuracy. Second best method is μCoMBo, which is encouraging since it means that adding the cooperation between the views leads to good results while promoting equity among imbalanced classes. The difference between early CoMBo and μCoMBo relies on the fact that the latter encourages such a cooperation among views, while the former only learns a model after the normalized concatenation of all the views. It is then worth noticing that multi-view learning (μCoMBo) actually helps to achieve better results when minimizing a bound of the confusion norm when facing class-imbalanced datasets.

Concerning the per-class recalls, early μCoMBo, Carako, and the multi-view μCoMBo all tend to reduce the impact of majority classes, while focusing on the minority ones. This behavior was expected through the minimization of the confusion norm. Symmetrically, AdaBoost.MM (both early and late fusion) and MKL clearly all favor the majority classes over the minority ones.

About measures dedicated to multi-class approaches, G-mean, and MAUC point out the smoothing effect of μCoMBo on errors which helps to better take into consideration the minority classes. According to G-mean and MAUC, the results strongly suggest that the best method is μCoMBo, which was expected since μCoMBo was designed to deal with imbalanced multi-view datasets. As for accuracy, the performances of the various methods on the G-mean imply that both μCoMBo and Carako promote some sort of leveling process among the classes, thus a better equity among them; however, μCoMBo ends up with better results than Carako, thanks to the cooperation among views. AdaBoost.MM and MKL have poor G-mean since they fail to recognize *beaver*, the minority class.

7.7 Discussion

This work merges imbalanced multi-view and multi-class learning, and proposes a boosting-like algorithm to address it. As far as we know, albeit many even recent results about ensemble-based imbalanced multi-class learning have been published [3, 5, 7, 9, 22, 23], no other approach has emerged that would meanwhile consider the capabilities of multi-view diversity of information sources. As a consequence, the proposed approach here is quite original, and cannot be fairly compared with any other state-of-the-art theory or algorithm.

Our preliminary experimental results show that our approach seems to be relevant for facing imbalanced views and classes, together with theoretical guarantees. As such, this work raises several questions and prospect works.

On the Confusion Matrix In Eqs. (7.4)–(7.6), the operator norm of the confusion matrix is bounded by the $l1$-norm. First, remark that due to the equivalence between norms in finite dimension, minimizing the (entry-wise) $l1$-norm is a viable alternative to the original goal, and it ensures that the learning procedure outputs

a classifier with a low error. Second, as briefly commented upon in Sect. 7.3, it is quite difficult to find an analytical expression for the operator norm of the confusion matrix in the supervised setting as given in Definition 1. In order to bypass these shortcomings and to tackle the original goal (minimizing the operator norm of the confusion matrix), future works will be focused on exploring alternative definitions for the confusion matrix, such as loss-based confusion matrices as in [17], entry-wise decomposition of the confusion matrix as in [15], and three-dimensional tensors.

On the Optimized Norm Aside from the choice of the confusion matrix, another research question touched upon in this paper is the choice of the norm. We think that it would be interesting to define and study other norms, such as the $l1$-norm, other p-norms, or even more exotic norms. In particular, a challenging problem is the definition of confusion matrices and confusion matrices' norms for the *multi-view setting* either as a generalization of the usual definitions to the three-dimensional tensor, or based on the tensor's theory.

On the Loss Functions The main advantage of the result in Eq. (7.6) is the flexibility of the loss functions. Although our work is mainly based on the exponential losses, Eqs. (7.9) and (7.15) show that other information can be embedded in the losses. As such, Eq. (7.6) can be used to derive other (novel) multi-view imbalanced learning methods by either choosing other loss functions, or modifying the information contained therein (such as enforcing the cooperation between the views, embedding prior information on the classes, etc.).

On the Combined Learners In Sect. 7.4, we argued that the main advantage of Carako (Algorithm 1) over MKL is the fact that our method is not limited to kernel methods. Other, more empirical, works will be focused on testing Carako with other learning methods and studying the effect that the cooperation between the views has on the final combined classifier.

On Theoretical Improvements Finally, future work will also be focused on finding tighter bounds for the result given in Eq. (7.15). As is, the constant term (right-handed side of the equation) depends on the number of classes and when this number is important, the constant might overshadow the true objective in the left-handed side of the equation. Although this might not present a real challenge for current multi-view imbalanced datasets, we think that finding tighter bounds will not only address a crucial issue for our approach, but it might also allow to derive novel algorithms in the same spirit as μCoMBo.

7.8 Conclusion

In this paper, we proposed various multi-view ensemble learning methods, proposed in Sects. 7.4 and 7.5, for dealing with imbalanced views and classes. The novelty of our approach consists in injecting a cooperation-based multi-view classifier (Eq. (7.7)) in the imbalanced classes framework (Eq. (7.5)). This choice is mainly

motivated by the promotion of the cooperation between the views in the output space, so that each view is associated with the classes it recognizes best. Our intuition is further confirmed by the empirical results in Sect. 7.6. We think that the work presented here is a first clear answer to the question posed in the introduction, while at the same time raising various research questions (e.g., the choices of the confusion matrix, its norm, the multi-view classifier, etc.). In the near future, a deep study of the complexity of μCoMBo is required, which mainly involves the specific properties of the non-linear convex optimization it relies on.

Acknowledgement This work is partially funded by the French ANR project LIVES ANR-15-CE23-0026.

References

1. Ayache, S., Quénot, G., Gensel, J.: Classifier fusion for SVM-based multimedia semantic indexing. In: ECIR, pp. 494–504 (2007)
2. Bach, F.R., Lanckriet, G.R.G.: Multiple kernel learning, conic duality, and the SMO algorithm. In: Proceedings of the 21st International Conference on Machine Learning (ICML) (2004)
3. Bi, J., Zhang, C.: An empirical comparison on state-of-the-art multi-class imbalance learning algorithms and a new diversified ensemble learning scheme. Knowl.-Based Syst. (2018). https://doi.org/10.1016/j.knosys.2018.05.037
4. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. J. Artif. Int. Res. **16**(1), 321–357 (2002)
5. Fernández, A., Garcia, S., Herrera, F., Chawla, N.V.: Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. J. Artif. Intell. Res. **61**, 863–905 (2018)
6. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proceedings of the International Conference on Machine Learning, pp. 148–156 (1996)
7. García, S., Zhang, Z.L., Altalhi, A., Alshomrani, S., Herrera, F.: Dynamic ensemble selection for multi-class imbalanced datasets. Inf. Sci. **445–446**, 22–37 (2018)
8. Gehler, P., Nowozin, S.: On feature combination for multiclass object classification. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 221–228 (2009)
9. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., Bing, G.: Learning from class-imbalanced data: review of methods and applications. Expert Syst. Appl. **73**, 220–239 (2017)
10. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. **21**(9), 1263–1284 (2009)
11. Huusari, R., Kadri, H., Capponi, C.: Multiview metric learning in vector-valued kernel spaces. In: Aistats, 2018, Lanzarote (2018)
12. Kadri, H., Ayache, S., Capponi, C., Koço, S., Dupé, F.X., Morvant, E.: The multi-task learning view of multimodal data. In: Asian Conference on Machine Learning, JMLR, pp. 261–276 (2013)
13. Koço, S., Capponi, C.: A boosting approach to multiview classification with cooperation. In: European Conference on Machine Learning (ECML), vol. 6912, pp. 209–228 (2011)
14. Lampert, C.H., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: IEEE Conference on Computer Vision and Pattern Recognition, 2009 (CVPR 2009), pp. 951–958 (2009)
15. Morvant, E., Koço, S., Ralaivola, L.: PAC-Bayesian generalization bound on confusion matrix for multi-class classification. In: International Conference on Machine Learning, pp. 815–822 (2012)

16. Mukherjee, I., Schapire, R.E.: A theory of multiclass boosting. J. Mach. Learn. Res. **14**(1), 437–497 (2013)
17. Ralaivola, L.: Confusion-based online learning and a passive-aggressive scheme. In: Neural Information Processing Systems Conference (2012)
18. Snoek, C., Worring, M., Smeulders, A.: Early versus late fusion in semantic video analysis. In: Proceedings of the 13th Annual ACM International Conference on Multimedia (MULTIMEDIA '05), pp. 399–402. ACM, New York (2005)
19. Sonnenburg, S., Raetsch, G., Henschel, S., Widmer, C., Behr, J., Zien, A., de Bona, F., Binder, A., Gehl, C., Franc, V.: The shogun machine learning toolbox. J. Mach. Learn. Res. **11**, 1799–1802 (2010)
20. Sun, Y., Kamel, M., Wang, Y.: Boosting for learning multiple classes with imbalanced class distribution. In: 2006 IEEE International Conference on Data Mining, HongKong, pp. 592–602 (2006)
21. Wang, S., Yao, X.: Multiclass imbalance problems: analysis and potential solutions. IEEE Trans. Syst. Man Cybern. B Cybern. **42**(4), 1119–1130 (2012)
22. Wu, F., Jing, X.Y., Shan, S., Zuo, W., Yang, J.Y.: Multiset feature learning for highly imbalanced data classification. In: Proceedings of AAAI, pp. 1583–1589 (2017)
23. Yijing, L., Haixiang, G., Xiao, L., Yanan, L., Jinling, L.: Adapted ensemble classification algorithm based on multiple classifier system and feature selection for classifying multi-class imbalanced data. Knowl.-Based Syst. **94**, 88–104 (2016)

Chapter 8
Entity Linking in Enterprise Search: Combining Textual and Structural Information

Sumit Bhatia

Abstract Fast and correct identification of named entities in queries is crucial for query understanding and to map the query to information in structured knowledge base. Most of the existing works have focused on utilizing search logs and manually curated knowledge bases for entity linking and often involve complex graph operations and are generally slow. We describe a simple, yet fast and accurate, probabilistic entity linking algorithm that can be used in enterprise settings where automatically constructed, domain-specific knowledge graphs are used. In addition to the linked graph structure, textual evidence from the domain-specific corpus is also utilized to improve the performance.

8.1 Introduction

With increasing popularity of virtual assistants like SIRI and Google Now, users are interacting with search systems by asking natural language questions that often contain named entity mentions. A large-scale study by Pang and Kumar [40] observed statistically significant temporal increases in the fraction of questions–queries received by search engines and searchers tend to use more question–queries for complex information needs [3]. In case of web search engines, a large fraction of queries contain a named entity (estimates vary from 40% [31] to 60% [42]). Hence, *fast* and *correct* identification of named entities in user queries is crucial for query understanding and to map the query to information in structured knowledge base. Advancements in semantic search technology have enabled modern information retrieval systems to utilize structured knowledge bases such as DBPedia [2] and Yago [45] to satisfy users' information needs.

Most of the existing works on entity linking focus on linking the entities in long documents [26, 30]. These methods make use of the large context around the target

S. Bhatia (✉)
IBM Research AI, New Delhi, India
e-mail: sumitbhatia@in.ibm.com

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_8

mention in the document. Therefore, these methods are limited to perform on long text documents. However, some methods have been proposed that perform entity linking in short sentences [20, 27]. They rely on the collective disambiguation [15] of all the entity mentions appear in the sentences. Thus, these methods take long time in computing the confidence scores for all the combinations.

Most of the existing work on entity linking in search queries utilizes information derived from query logs and open knowledge bases such as DBPedia and Freebase (Sect. 8.2). Such techniques, however, are not suited for enterprise and domain-specific search systems such as legal, medical, and healthcare, due to very small user bases resulting in small query logs and the absence of rich domain-specific knowledge bases. Recently, there have been development of systems for automatic construction of semantic knowledge bases for domain-specific corpora [12, 48] and systems that use such domain-specific knowledge bases [38]. In this chapter, we describe a method for entity disambiguation and linking, developed especially for enterprise settings, where such external resources are often not available. The proposed system offers users a search interface to search for the indexed information and uses the underlying knowledge base to enhance search results and provide additional entity-centric data exploration capabilities that allow users to explore hidden relationships between entities discovered automatically from a domain-specific corpus.

The system *automatically* constructs a structured knowledge base by identifying entities and their relationships from input text corpora using the method described by Castelli et al. [12]. Thus, for each relationship discovered by the system, the corresponding mention text provides additional contextual information about the entities and relationships present in that mention. We posit that the *dense graph structure* discovered from the corpus, as well as the *additional context provided by the associated mention text*, can be utilized together for linking entity name mentions in search queries to corresponding entities in the graph. Our proposed entity linking algorithm is intuitive, relies on a theoretical sound probabilistic framework, and is fast and scalable with an average response time of $\approx$ **87** ms. Figure 8.1 shows the working of proposed algorithm in action where top ranked

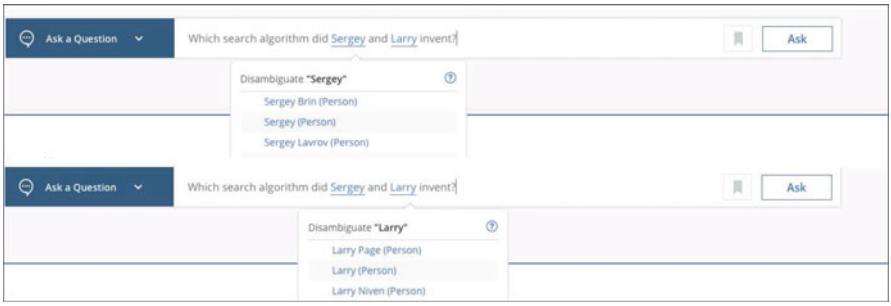


Fig. 8.1 Entity suggestions produced by proposed approach using text and entity context in search query

suggestions for named mentions `Sergey` and `Larry` are showed. As will be described in detail in Sect. 8.3, note that the algorithm is making these suggestions by utilizing the terms in questions (search, algorithm) as well as relationships between all target entities for mentions "Sergey" and "Larry" in the graph. The algorithm figures out that entities "Sergey Brin" and "Larry Page" have strong evidences from their textual content as well as these two entities are strongly connected in the graph, and hence they are suggested as most probable relevant entities in the context of question.

The material presented in this chapter is an extended version of our ESWC 2016 paper [5], and we provide a detailed survey of the representative work on entity linking and discuss their shortcomings when applied to enterprise settings. We describe our proposed approach in detail with several examples to illustrate the working of the algorithm. We hope that the additional details will help the readers, especially beginners and practitioners, to understand the finer details and workings of the proposed approach and will help them implement the approach for their custom applications.

8.2 Related Work

We first discuss early works that provide the foundation for the general entity linking task and define the problem in context of knowledge graphs. We then review representative works that addressed entity linking in longer documents as well as much shorter text fragments such as web queries and tweets.

8.2.1 Entity Linking Background

At its core, the problem of *entity linking* is similar to the general problem of *record linkage* that was first introduced by Dunn [18] in the context of assembling all public records of an individual. This idea was further popularized by Newcombe et al. [39] that proposed the use of computers to link multiple separately recorded pieces of information about a particular person or family for census applications. In general, *record linkage* refers to the task of finding and linking records about an entity spread across multiple datasets. This is an extensively studied problem in the field of databases and data mining, and a detailed survey is out of the scope of this chapter. We direct the interested reader to excellent surveys on this topic by Brizan and Tansel [11] and Christen [14].

Entity linking, as studied in this chapter, refers to the task of linking the mention of a named entity in text (a sentence, keyword query, etc.) to the corresponding entity in a knowledge base. Let us consider the following piece of text about Barack Obama to understand the challenges involved.

Barack Obama served as the 44^{th} President of the United States from 2009 to 2017. *He* was born in Honolulu, Hawaii. *Obama* has two daughters.

A named entity recognizer [35, 37] when run on the above text will be able to identify *Barack Obama* and *Obama* as two named entities. However, these two different *surface forms* correspond to the same entity $BarackObama$ in the underlying knowledge base. Hence, it is required for the system to be able to identify that these two different mentions are variations of the same entity name and link them to the same canonical entity, a task known as *entity normalization* [29]. Also, note that in the above example text, the pronoun *he* also refers to Barack Obama. This task of determining different expressions (nouns, pronouns, etc.) that refer to the same entity is known as *coreference resolution* [19]. Note that depending upon the requirements, it may also be required to perform coreference resolution and entity normalization across multiple documents [4, 28, 41]. While the tasks of named entity recognition, coreference resolution, and entity normalization have been studied extensively, entity linking involves the additional step of aligning the identified and normalized entity mention to its corresponding entity in the knowledge base.

8.2.2 Linking Entities in Documents and Web Pages

Entity linking has been studied under various application scenarios. SemTag [17] was one of the first systems to consider the task of linking entities in web pages to entities in a knowledge base (Stanford TAP entity catalog [22]). Wikipedia, owing to exhaustive coverage of general concepts, has been used as the underlying knowledge base to link entity mentions in documents, web pages, news articles, etc. [15, 26, 34, 36, 43]. Mihalcea and Csomai [34] introduced the *Wikify!* system to extract keywords from documents and link them to their corresponding Wikipedia pages. Cucerzan [15] utilized Wikipedia category information, in addition to contextual similarities between documents and Wikipedia based features entity normalization and linking. Kulkarni et al. [30] premised that entities mentioned in a coherent document are topically related and showed that utilizing this information to collectively link entities in a document can help improve performance. Hoffart et al. [26] proposed a comprehensive framework for collective entity disambiguation and linking that combines local contextual information about the input mention with coherence among candidate entities for all entity mentions together.

8.2.3 Linking Entities in Short Text Fragments

The methods discussed till now have focused on performing entity linking for longer documents like web pages, news articles, etc. Such documents generally contain enough contextual clues as well as additional metadata that could aid identifying

appropriate mentions. In case of shorter text documents, such as microblogs, or web search queries that are generally a few keywords long, successful entity linking has to rely on specific application specific contextual clues and metadata in absence of large document context. For example, in case of linking entity mentions in tweets, user characteristics, interest profiles, social network properties such as retweets and likes can be utilized [23, 32, 44]. Ferragina and Scaiella [20] utilize the anchor texts of Wikipedia articles to construct a dictionary of different surface forms or name variations for entities and use that to identify entity mentions in short text fragments. The final set of target entities is then determined by collective agreement among different potential mappings. Hoffart et al. [27] describe an algorithm that performs collective entity linking by computing overlap between the sets of keywords associated with each target entity. For creating the set of keywords, noun phrases from Wikipedia entity pages are used. The proposed algorithm achieves good performance for both short and long texts, as well as for less popular, long tail entities.

Another challenging setting for performing entity linking is in the context of web search queries that are often just a collection of few keywords. Typical ways to perform entity linking in such systems is to approximate semantic similarity between queries and entities by utilizing their respective language models [21, 25]. Successful identification and linking of entity mentions in queries can also help improve retrieval performance by means of query expansion using entity features from the knowledge base [16]. Another challenge for entity linking in search systems is that it has to be performed before the actual retrieval takes place and thus, needs to be completed in just a few milliseconds. Blanco, Ottaviano, and Meij [10] describe a space efficient algorithm for entity linking for web search queries that is able to process queries in sub-milliseconds time.

These methods use features derived from query logs to gather user context, target documents, etc., to get context. However, in many enterprise systems, such additional metadata is not readily available [7]. Further, the knowledge bases used in such systems may not be as rich as Wikipedia lacking hyperlinks, metadata, etc., and are often constructed using automated methods [8]. However, context is important [6]. In this work, we discuss how we can utilize the limited context available in the input query (text, entity mentions) and utilize the textual information in background corpus coupled with rich graph structure to perform entity linking in enterprise search systems.

8.3 Proposed Approach

We first describe the problem setting and our assumptions, and provide a probabilistic formulation of the entity linking problem. We also discuss how different application settings can be mapped to the proposed formulation and then provide a solution for entity linking that utilizes structural properties of entities in the knowledge graph and information from the background text corpus.

8.3.1 Problem Setting

Let us consider a knowledge graph $\mathcal{K} = \{\mathcal{E}, \mathcal{R}\}$ where $\mathcal{E}$ is the set of entities (nodes) and $\mathcal{R}$ is the set of relationships (edges). Let us also assume the availability of a background text corpus $\mathcal{C}$.[1] Let $\mathcal{M}_r$ be the set of all the mentions of the relationship r in the background text corpus. As an example, consider the relationship $\langle SteveJobs \rangle$, $founderOf$, $\langle AppleInc. \rangle$ and one of its many mentions from Wikipedia, *"Jobs and Wozniak co-founded Apple in 1976 to sell Wozniak's Apple I personal computer."* Note that in addition to the relationship under consideration, this mention also provides additional contextual clues about the entities $SteveJobs$ and $AppleInc$. (Wozniak, personal computer are related to Steve Jobs and Apple Inc.)

8.3.2 Problem Formulation

Let $Q = \{C, T\}$ be the input query where T is the ambiguous token, and $C = \{E_c, W_c\}$ is the context under which we have to disambiguate T. The context is provided by the words ($W_c = \{w_{c1}, w_{c2}, \ldots, w_{cl}\}$) in the query and the set of unambiguous entities $E_c = \{e_{c1}, e_{c2}, \ldots, e_{cm}\}$. Note that initially, this entity set can be empty if there are no unambiguous entity mentions in the query and in such cases, only textual information is considered. The task is to map the ambiguous token T to one of the possible target entities.

This is a generalized statement of the entity linking task and covers a variety of end-applications and scenarios as discussed below.

- **Search Systems:** The user typically enters a few keywords and the task is to link the keywords in query to an entity in the knowledge graph. Note that not all the terms in the query correspond to entity mentions and the problem is further exacerbated by the inherent ambiguity of keyword queries [24]. For example, in the query `obama speeches`, `obama` corresponds to the entity *Barack Obama* and `speeches` provides the information need of the user. Also note that keyword queries lack the additional contextual information that is present while linking entities in documents. To overcome this, web search systems often utilize query logs and user activity to gather context about users' information needs [24]. Once terms in the queries are linked to corresponding entities in the graph, related entities can also be offered as recommendations to the end-user for further browsing [9].

[1] For domain-specific applications where the knowledge graph is constructed using automated methods, the set of input documents constitute the background corpus. For applications that use generic, open-domain knowledge bases such as DBPedia and WikiData, Wikipedia could be used as the background text corpus.

- **Question Answering Systems:** By identifying entities of interest in the question, the underlying knowledge base can be used to retrieve the appropriate facts required to answer the question [47]. In a typical QA system, the user enters a natural language question such as `When did Steve become ceo of Microsoft?` Here, the terms of interest are `Steve` and `Microsoft`. Also note that in this example, `Microsoft` also provides contextual evidence that provides additional support for *Steve Ballmer* compared to many other person entities named Steve such *Steve Jobs* or *Steve Wozniak* that will have less relevance to `Microsoft` than *Steve Ballmer*. Once the system correctly links `steve` to *Steve Ballmer*, appropriate facts from the knowledge graph can be easily retrieved and presented as answer to the user.

8.3.3 Proposed Solution

On receiving the input query, the first step in the solution to the problem as formulated above is to identify entity mentions in the query. These mentions are then linked to the corresponding entity in the knowledge graph. These entity mentions could be identified using NLP libraries such as Apache Open NLP[2] and Stanford Named Entity Recognizer.[3]

After identifying the token T that is a named entity mention in the query Q, the next step is to generate a list of target candidate entities. Such a list could be generated by using a dictionary that contains different surface forms of the entity names [30, 46, 49, 50]. For example, a dictionary could be constructed that maps different surface forms of the entity *Barack Obama* such as `Barack Obama, Barack H. Obama, and President Obama` to the entity. Since we are interested in mapping the token to entities in the knowledge graph $\mathcal{K} = \{\mathcal{E}, \mathcal{R}\}$, we select all the entities that contain token T as a sub-string in their name. For example, for the token `Steve` all entities such as `Steve Jobs, Steve Wozniak, and Steve Lawrence` constitute the set of target entities. Note that for domain-specific applications, such a dictionary could also be constructed by using domain-specific sources such as the gene and protein dictionaries used in the KnIT system for studying protein–protein interactions [38]. For generic, open-domain systems, Wikipedia has been used extensively to create such dictionaries by utilizing disambiguation and redirect pages, anchor text and hyperlinks, etc.

Formally, let $E_T = \{e_{T1}, e_{T2}, \ldots, e_{Tm}\}$ be the set of target entities for the ambiguous token T in the query. Using the context information, we can produce a ranked list of target entities by computing $P(e_{Ti}|C)$, i.e., the probability that the user is interested in entity e_{Ti} given the context C. Using Bayes' theorem, we can write $P(e_{Ti}|C)$ as follows:

[2] http://opennlp.apache.org/.

[3] https://nlp.stanford.edu/software/CRF-NER.html.

$$P(e_{Ti}|C) = \frac{P(e_{Ti})P(C|e_{Ti})}{P(C)} \tag{8.1}$$

Here $P(e_{Ti})$ represents the prior probability of the entity e_{Ti} to be relevant without any context information. This prior probability can be computed in multiple ways based on the application requirements. For example, priors can be computed based on frequency of individual entities or temporal information (such as recency) in case of news domain. In this work, we assume a frequency based prior indicating that in the absence of any context information, the probability of an entity being relevant is directly proportional to its frequency in the graph. Further, since we are only interested in relative ordering of the target entities, we can ignore the denominator $P(C)$ as its value will be same for all the target entities. With these assumptions, Eq. (8.1) can be re-written as follows:

$$P(e_{Ti}|C) \propto P(e_{Ti}) \times P(C|e_{Ti}) \tag{8.2}$$

Here $P(C|e_{Ti})$ represents the probability of observing the context C after having seen the entity e_{Ti}. Note that the context C consists of two components—text context and entity context. Assuming that the probability of observing text and entity context is conditionally independent, above equation can be reduced as follows:

$$P(e_{Ti}|C) \propto P(e_{Ti}) \times P(W_c|e_{Ti}) \times P(E_c|e_{Ti}) \tag{8.3}$$

$$= \underbrace{P(e_{Ti})}_{\text{entity prior}} \times \underbrace{\prod_{w_c \in W_c} P(w_c|e_{Ti})}_{\text{text context}} \times \underbrace{\prod_{e_c \in E_c} P(e_c|e_{Ti})}_{\text{entity context}} \tag{8.4}$$

8.3.3.1 Computing Entity Context Contribution

The *entity context* factor in Eq. (8.4) corresponds to the evidence for target entity given E_c, the set of entities forming the context. For each individual entity e_c forming the context, we need to compute $P(e_c|e_{Ti})$, i.e., the probability of observing e_c after observing the target entity e_{Ti}. Intuitively, there is a higher chance of observing an entity that is involved in multiple relationship with e_{Ti} than an entity that only has a few relationships with e_{Ti}. Thus, we can estimate $P(e_c|e_{Ti})$ as follows:

$$P(e_c|e_{Ti}) = \frac{\text{relCount}(e_c, e_{Ti}) + 1}{\text{relCount}(e_c) + |E|} \tag{8.5}$$

Note that the factor of 1 in numerator and $|E|$ (size of entity set E) in the denominator have been added to smoothen the probability values for entities that are not involved in any relationship with e_{Ti}.

8.3.3.2 Computing Text Context Contribution

The *text context* factor in Eq. (8.4) corresponds to the evidence for target entity given W_c, the terms present in the input query. For each individual query term w_c, we need to compute $P(w_c|e_{Ti})$, i.e., the probability of observing w_c given e_{Ti}. In order to compute this probability, we construct *mention language models* for each entity in the knowledge graph that capture different contexts in which the entity appears in the corpus.

To construct such a mention language model for an entity e, we need to capture all the *mention sentences*, i.e., the sentences from the text corpus that talk about entity e. In automatically constructed graphs, where rule based or machine learned systems identify entity and relationship mentions from text, the source text for each extracted relationship and entity can be utilized to capture all the mention sentences for entity e by combining all the source sentences from which the entity and its relationships were identified. The *mention documents* created in this way capture different contexts under which the entity has been observed in the input corpus. For example, a lot of relationships of Steve Jobs are with Apple products, executives, etc. So sentences for these relationships will contain mentions of things related to Apple, in addition to entity names. For example, sentences containing relationships of Steve Jobs with iPhone will contain words like *design, touchscreen, mobile, apps, battery,* etc., and all these *contextual clues* are captured in *mention document* for Steve Jobs.

The mention documents created in this way can be used to compute the probability $P(w_c|e_{Ti})$ as follows:

$$P(w_c|E_{Ti}) = P(w_c|M_{E_{Ti}}) \tag{8.6}$$

$$= \frac{\text{no. of times } w_c \text{ appears in } M_{E_{Ti}} + 1}{|M_{E_{Ti}}| + N} \tag{8.7}$$

Here N is the size of the vocabulary and $M_{E_{Ti}}$ is the mention document for entity E_{Ti}.

8.3.3.3 Putting It All Together

We now illustrate the working of the proposed approach through an example as illustrated in Fig. 8.2. Consider the input question, "Which search algorithm did sergey and larry invent." In this question, the NER module identifies *sergey* and *larry* as the two named entities that need to be linked to the corresponding entities in the knowledge graph. The two ambiguous tokens and the natural language question are fed as input to the system. As discussed, the first step is to generate a list of target entities that is performed by retrieving all entities from the graph containing *sergey* and *larry* in their names. For each such target entity, we need to compute

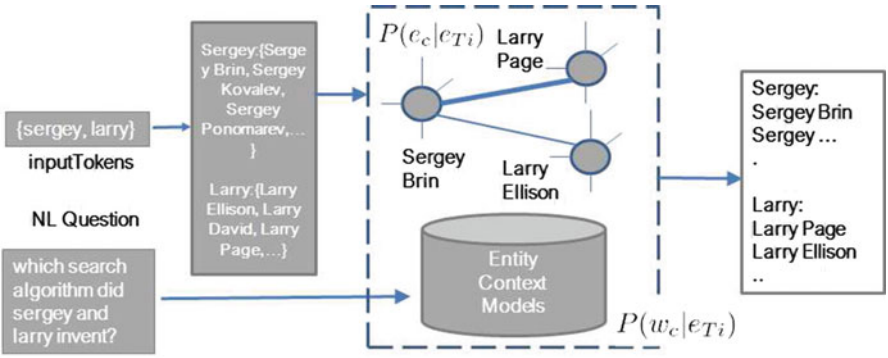


Fig. 8.2 Illustration of the proposed approach

the entity and text context components as described in Eq. (8.4). The entity context component helps in collective disambiguation of entities by taking into account the pairwise relevance of the target entities for the two ambiguous tokens. For example, the pair <Sergey Brin, Larry Page> will have much stronger connections in the graph (both Google co-founders share many common relations) compared to the pair <Sergey Brin, Larry Ellison> (Larry Ellison being co-founder of Oracle shares much less relations with Sergey Brin). Likewise, for the text context component, the mention language models of all target entities are used to find the entities that have the highest probability of generating the context terms in the questions such as *search and algorithm*. Thus, entities such as *Sergey Brin, Larry page, and Larry Ellison* get high text context component scores due to their relations with computer science related concepts. The two scores for all the target entities are combined to produce a final ranked list as illustrated in the figure.

8.4 Evaluation

8.4.1 Data Description

We use a semantic graph constructed from text of all articles in Wikipedia by automatically extracting the entities and their relations by using IBM's Watson natural language understanding (NLU) services.[4] Even though there exist popular knowledge bases like DBPedia that contain high quality data, we chose to construct a semantic graph using automated means as such a graph will be closer to many practical real-world scenarios where high quality curated graphs are often not available and one has to resort to automatic methods of constructing knowledge

[4] https://www.ibm.com/watson/services/natural-language-understanding/.

bases. Our graph contains more than 30 million entities and 192 million distinct relationships in comparison to 4.5 million entities and 70 million relationships in DBpedia.

8.4.2 Benchmark Test Set and Baselines

For evaluating the proposed approach, we use the KORE50 [27] dataset that contains 50 short sentences with highly ambiguous entity mentions (Table 8.1). This widely used dataset is considered among the hardest dataset for entity disambiguation and is being used widely for evaluating entity disambiguation/linking approaches. Further, on an average, there are only 14 words and roughly 3 mentions per sentence, thus making it ideal for evaluating our approach as it enables us to identify our interactive approach. Average sentence length (after stop word removal) is 6.88 words per sentence and each sentence has 2.96 entity mentions on an average. Every mention has an average of 631 candidates to disambiguate in YAGO knowledge base [45]. However, it varies for different knowledge bases. Our automatically constructed knowledge base has *2261 candidates per mention* to disambiguate illustrating the difficulty in entity linking due to high noise in automatically constructed knowledge bases when compared with manually curated/cleaned knowledge bases such as DBpedia. We also provide the performance numbers for a number of commonly used methods on the same dataset for reference [1, 13, 26, 27, 33] (Table 8.2).

Table 8.1 Characteristics of KORE50 dataset

Average sentence length	14.68
Average sentence length after stop word removal	6.88
Average entity mentions per sentence	2.96

Table 8.2 Entity disambiguation accuracy, measured in terms of precision, as achieved by the proposed approach

Method	Precision
Joint-DiSER-TopN [1]	0.72
AIDA-2012 [26]	0.57
AIDA-2013 [27]	0.64
Wikifier [13]	0.41
DBpedia spotlight [33]	0.35
Proposed method accuracy @ Rank 1	0.52
Proposed method accuracy @ Rank 5	0.65
Proposed method accuracy @ Rank 10	0.74

The table also provides accuracy achieved by several commonly used methods at Rank 1, as reported in the respective papers. For the proposed approach, precision achieved at Ranks 5 and 10 is also reported

8.4.3 Results and Discussions

The results of our proposed approach and various other state-of-the-art methods for entity linking on the same dataset are tabulated in Table 8.2. We note that on the standard KORE 50 dataset for entity disambiguation, our proposed approach, while being much simpler than the other reported methods, achieves comparable performance in terms of precision values at Rank 1. The top achieving methods do achieve better accuracy number but at the cost of higher complexity, reliance on many external resources of data, and consequently, slower speeds. For example, as reported by Hoffart et al. [27], the average time taken for disambiguation is 1.285 s with a standard deviation of 3.925 s. On the other hand, as we observe from Table 8.3, median response time for the proposed approach is about 86 ms, with the maximum response time being 125 ms. Such low response times were possible due to the fact that we utilized the signals from mention text and relationship information about entities that are much more computationally efficient to compute,[5] instead of performing complex and time-consuming graph operations as in other methods, while not sacrificing on the accuracy.

Figure 8.3 illustrates the working of proposed system in action for a variety of input <query,context> combinations. In Fig. 8.3a, the token `Steve` is provided without any context and the system returns a list sorted by entity prior (frequency). Next, in Fig. 8.3b–d, the results for the token `Steve` under different context terms are shown. Note how the system finds different entities in each case with changing context. Likewise, Fig. 8.3e shows the results for token `larry` without any context. However, as soon as we provide another token to disambiguate (`Sergey`) in Fig. 8.3f, the entity context component kicks in and collectively determines the most probable entities for both `sergey` and `larry`.

Table 8.3 Average candidate list size and response times per query

	Candidate size	Response time (ms)
Min.	0	85
Average	7917.27	87.34
Median	2261.5	86
Max.	183,546	125

The experiments were conducted on a standard MacBook Pro laptop with 16 GB RAM and an Intel i7 processor

[5] Text context components can be computed by using an inverted index implementation where using the context terms as queries, most relevant mention docs (and thus the corresponding entities) can be retrieved in a single query. Likewise, entity context component can be computed by just counting the number of connections between target entities—can be performed in a single optimized SQL query.

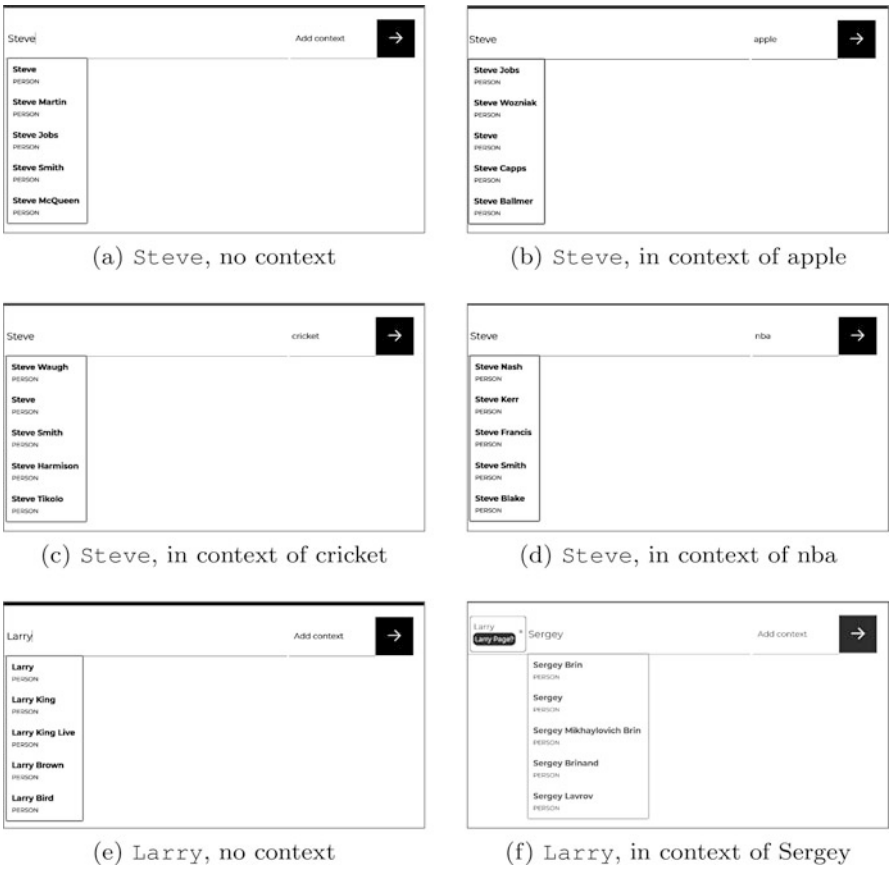


Fig. 8.3 Some examples of the proposed entity linking approach in action. Note how the suggestions for entities change in sub-figures (**a**)–(**d**) with varying contexts. Also note that how the entity context helps retrieve relevant results for `larry` in sub-figures (**e**) and (**f**). First, in sub-figure (**e**), in the absence of any context, the suggestions offered for `Larry` are simply ranked by the frequency prior, suggesting most popular entities containing `larry` in their name. Next, in sub-figure (**f**), when the user types `Sergey`, the system collectively disambiguates `Larry` as `Larry Page` and `Sergey` as `Sergey Brin`—note that this corresponds to the entity context component of the ranking function

8.5 Conclusions

In this chapter, we discussed the problem of mapping entity mentions in natural language search queries to corresponding entities in an automatically constructed knowledge graph. We provided a review of representative works on entity linking and their shortcomings when applied to enterprise settings. We then proposed an approach that utilizes the dense graph structure as well as additional context

provided by the mention text. Experimental evaluation on a standard dataset shows that the proposed approach is able to achieve high accuracy (comparable to other state-of-the-art methods) with a median response time of 86 ms.

References

1. Aggarwal, N., Buitelaar, P.: Wikipedia-based distributional semantics for entity relatedness. In: 2014 AAAI Fall Symposium Series (2014)
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: A nucleus for a web of open data. In: Aberer, K., Choi, K.S., Noy, N.F., Allemang, D., Lee, K.I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) 6th International Semantic Web Conference (ISWC 2007). Lecture Notes in Computer Science, vol. 4825, pp. 722–735. Busan (2007). https://doi.org/10.1007/978-3-540-76298-0_52
3. Aula, A., Khan, R.M., Guan, Z.: How does search behavior change as search becomes more difficult? In: Mynatt, E.D., Schoner, D., Fitzpatrick, G., Hudson, S.E., Edwards, W.K., Rodden, T. (eds.) Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, Atlanta, GA, 10–15 April 2010, pp. 35–44. Association for Computing Machinery, New York (2010). http://doi.acm.org/10.1145/1753326.1753333
4. Bagga, A., Baldwin, B.: Entity-based cross-document coreferencing using the vector space model. In: Boitet, C., Whitelock, P. (eds.) ACL/COLING, pp. 79–85. Morgan Kaufmann Publishers/ACL (1998). http://aclweb.org/anthology/P/P98/
5. Bhatia, S., Jain, A.: Context sensitive entity linking of search queries in enterprise knowledge graphs. In: Sack, H., Rizzo, G., Steinmetz, N., Mladenic, D., Auer, S., Lange, C. (eds.) The Semantic Web – ESWC 2016 Satellite Events, Heraklion, Crete, 29 May–2 June 2016, Revised Selected Papers. Lecture Notes in Computer Science, vol. 9989, pp. 50–54 (2016). https://doi.org/10.1007/978-3-319-47602-5_11
6. Bhatia, S., Vishwakarma, H.: Know Thy Neighbors, and More! Studying the Role of Context in Entity Recommendation. In: HT '18: 29th ACM Conference on Hypertext and Social Media, 9–12 July 2018, Baltimore. Association for Computing Machinery, New York (2018). https://doi.org/10.1145/3209542.3209548
7. Bhatia, S., Majumdar, D., Mitra, P.: Query suggestions in the absence of query logs. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '11, pp. 795–804. Association for Computing Machinery, New York (2011). http://doi.acm.org/10.1145/2009916.2010023
8. Bhatia, S., Rajshree, N., Jain, A., Aggarwal, N.: Tools and infrastructure for supporting enterprise knowledge graphs. In: Cong, G., Peng, W., Zhang, W.E., Li, C., Sun, A. (eds.) Proceedings of the 13th International Conference Advanced Data Mining and Applications, ADMA 2017, Singapore, 5–6 November 2017. Lecture Notes in Computer Science, vol. 10604, pp. 846–852. Springer, Berlin (2017). https://doi.org/10.1007/978-3-319-69179-4_60
9. Blanco, R., Cambazoglu, B.B., Mika, P., Torzec, N.: Entity recommendations in web search. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) The Semantic Web – ISWC 2013, pp. 33–48. Springer, Berlin (2013)
10. Blanco, R., Ottaviano, G., Meij, E.: Fast and space-efficient entity linking for queries. In: Cheng, X., Li, H., Gabrilovich, E., Tang, J. (eds.) Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, Shanghai, 2–6 February 2015, pp. 179–188. Association for Computing Machinery, New York (2015). http://dl.acm.org/citation.cfm?id=2684822

11. Brizan, D.G., Tansel, A.U.: A. survey of entity resolution and record linkage methodologies. Commun. IIMA **6**(3), 5 (2006)
12. Castelli, V., Raghavan, H., Florian, R., Han, D.J., Luo, X., Roukos, S.: Distilling and exploring nuggets from a corpus. In: SIGIR, pp. 1006–1006 (2012)
13. Cheng, X., Roth, D.: Relational inference for wikification. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1787–1796. Association for Computational Linguistics, Seattle (2013). http://aclweb.org/anthology/D/D13/D13-1184.pdf
14. Christen, P.: Data Matching – Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Data-Centric Systems and Applications. Springer, Berlin (2012)
15. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. In: Eisner, J. (ed.) EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 28–30 June 2007, Prague, pp. 708–716. Association for Computational Linguistics, Seattle (2007). http://www.aclweb.org/anthology/K/K07/
16. Dalton, J., Dietz, L., Allan, J.: Entity query feature expansion using knowledge base links. In: Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR '14, pp. 365–374. Association for Computing Machinery, New York (2014). http://doi.acm.org/10.1145/2600428.2609628
17. Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., Rajagopalan, S., Tomkins, A., Tomlin, J.A., Zien, J.Y.: Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In: Proceedings of the 12th International Conference on World Wide Web, WWW '03, pp. 178–186. Association for Computing Machinery, New York (2003). http://doi.acm.org/10.1145/775152.775178
18. Dunn, H.L.: Record linkage. Am. J. Public Health and the Nations Health **36**(12), 1412–1416 (1946). https://doi.org/10.2105/AJPH.36.12.1412. PMID: 18016455
19. Elango, P.: Coreference resolution: A survey. Technical Report, University of Wisconsin, Madison, WI (2005)
20. Ferragina, P., Scaiella, U.: Fast and accurate annotation of short texts with Wikipedia pages. IEEE Softw. **29**(1), 70–75 (2012). http://dx.doi.org/10.1109/MS.2011.122
21. Gottipati, S., Jiang, J.: Linking entities to a knowledge base with query expansion. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11, pp. 804–813. Association for Computational Linguistics, Stroudsburg (2011). http://dl.acm.org/citation.cfm?id=2145432.2145523
22. Guha, R., McCool, R.: Tap: a semantic web test-bed. Web Semant. Sci. Serv. Agents on the World Wide Web **1**(1), 81–87 (2003). https://doi.org/10.1016/j.websem.2003.07.004. http://www.sciencedirect.com/science/article/pii/S1570826803000064
23. Guo, S., Chang, M.W., Kiciman, E.: To link or not to link? a study on end-to-end tweet entity linking. In: Vanderwende, L., III, H.D., Kirchhoff, K. (eds.) Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, 9–14 June 2013, Westin Peachtree Plaza Hotel, Atlanta, pp. 1020–1030. The Association for Computational Linguistics (2013). http://aclweb.org/anthology/N/N13/N13-1122.pdf
24. Hasibi, F., Balog, K., Bratsberg, S.E.: Entity linking in queries: tasks and evaluation. In: Proceedings of the 2015 International Conference on The Theory of Information Retrieval, ICTIR '15, pp. 171–180. Association for Computing Machinery, New York (2015). http://doi.acm.org/10.1145/2808194.2809473
25. Hasibi, F., Balog, K., Bratsberg, S.E.: Entity linking in queries: efficiency vs. effectiveness. In: Jose, J.M., Hauff, C., Altingövde, I.S., Song, D., Albakour, D., Watt, S.N.K., Tait, J. (eds.) Proceedings of the 39th European Conference on IR Research Advances in Information Retrieval, ECIR 2017, Aberdeen, 8–13 April 2017. Lecture Notes in Computer Science, vol. 10193, pp. 40–53 (2017)

26. Hoffart, J., Yosef, M.A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., Weikum, G.: Robust disambiguation of named entities in text. In: EMNLP, pp. 782–792. Association for Computational Linguistics, Seattle (2011). http://www.aclweb.org/anthology/D11-1072
27. Hoffart, J., Seufert, S., Nguyen, D.B., Theobald, M., Weikum, G.: Kore: keyphrase overlap relatedness for entity disambiguation. In: Chen, X.-W., Lebanon, G., Wang, H., Zaki, M.J. (eds.) 21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, 29 October–02 November 2012, pp. 545–554. Association for Computing Machinery, New York (2012). http://dl.acm.org/citation.cfm?id=2396761
28. Huang, J., Treeratpituk, P., Taylor, S.M., Giles, C.L.: Enhancing cross document coreference of web documents with context similarity and very large scale text categorization. In: Huang, C.R., Jurafsky, D. (eds.) COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23–27 August 2010, Beijing, pp. 483–491. Tsinghua University Press (2010). http://aclweb.org/anthology/C/C10/
29. Khalid, M.A., Jijkoun, V., de Rijke, M.: The impact of named entity normalization on information retrieval for question answering. Springer, New York (2009). http://dare.uva.nl/record/297954
30. Kulkarni, S., 0003, A.S., Ramakrishnan, G., Chakrabarti, S.: Collective annotation of Wikipedia entities in web text. In: IV, J.F.E., Fogelman-Soulié, F., Flach, P.A., Zaki, M.J. (eds.) Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, 28 June–1 July, 2009, pp. 457–466. Association for Computing Machinery, New York (2009). http://doi.acm.org/10.1145/1557019.1557073
31. Lin, T., Pantel, P., Gamon, M., Kannan, A., Fuxman, A.: Active objects: Actions for entity-centric search. In: World Wide Web. Association for Computing Machinery, New York (2012). http://research.microsoft.com/apps/pubs/default.aspx?id=161389
32. Liu, X., Li, Y., Wu, H., Zhou, M., Wei, F., Lu, Y.: Entity linking for tweets. In: ACL (1), pp. 1304–1311. The Association for Computer Linguistics (2013). http://aclweb.org/anthology/P/P13/
33. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: DBpedia spotlight: shedding light on the web of documents. In: Proceedings of the 7th International Conference on Semantic Systems, pp. 1–8. Association for Computing Machinery, New York (2011)
34. Mihalcea, R., Csomai, A.: Wikify!: linking documents to encyclopedic knowledge. In: Silva, M.J., Laender, A.H.F., Baeza-Yates, R.A., McGuinness, D.L., Olstad, B., Olsen, Ø.H., Falcão, A.O. (eds.) Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, 6–10 November 2007, pp. 233–242. Association for Computing Machinery, New York (2007). http://doi.acm.org/10.1145/1321440.1321475
35. Mohit, B.: Named entity recognition, pp. 221–245 (2014). https://doi.org/10.1007/978-3-642-45358-8_7
36. Moro, A., Raganato, A., Navigli, R.: Entity linking meets word sense disambiguation: a unified approach. Trans. Assoc. Comput. Linguist. **2**, 231–244 (2014). https://transacl.org/ojs/index.php/tacl/article/view/291
37. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticæ Investigationes **30**(1), 3–26 (2007). http://www.jbe-platform.com/content/journals/10.1075/li.30.1.03nad
38. Nagarajan, M., Wilkins, A.D., Bachman, B.J., Novikov, I.B., Bao, S., Haas, P.J., Terrón-Díaz, M.E., Bhatia, S., Adikesavan, A.K., Labrie, J.J., Regenbogen, S., Buchovecky, C.M., Pickering, C.R., Kato, L., Lisewski, A.M., Lelescu, A., Zhang, H., Boyer, S., Weber, G., Chen, Y., Donehower, L.A., Spangler, W.S., Lichtarge, O.: Predicting future scientific discoveries based on a networked analysis of the past literature. In: Cao, L., Zhang, C., Joachims, T., Webb, G.I., Margineantu, D.D., Williams, G. (eds.) Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, 10–13 August 2015, pp. 2019–2028. Association for Computing Machinery, New York (2015). http://dl.acm.org/citation.cfm?id=2783258

39. Newcombe, H.B., Kennedy, J.M., Axford, S.J., James, A.P.: Automatic linkage of vital records. Science **130**(3381), 954–959 (1959). http://science.sciencemag.org/content/130/3381/954
40. Pang, B., Kumar, R.: Search in the lost sense of "query": question formulation in web search queries and its temporal changes. In: ACL (Short Papers), pp. 135–140. The Association for Computer Linguistics (2011). http://www.aclweb.org/anthology/P11-2024
41. Popescu, O.: Dynamic parameters for cross document coreference. In: Huang, C.R., Jurafsky, D. (eds.) COLING 2010, 23rd International Conference on Computational Linguistics, Posters Volume, 23–27 August 2010, Beijing, pp. 988–996. Chinese Information Processing Society of China (2010). http://aclweb.org/anthology/C/C10/C10-2114.pdf
42. Pound, J., Mika, P., Zaragoza, H.: Ad-hoc object retrieval in the web of data. In: Proceedings of the 19th International Conference on World Wide Web, WWW '10, pp. 771–780. Association for Computing Machinery, New York (2010). http://doi.acm.org/10.1145/1772690.1772769
43. Ratinov, L., Roth, D., Downey, D., Anderson, M.: Local and global algorithms for disambiguation to Wikipedia. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, HLT '11, vol. 1, pp. 1375–1384. Association for Computational Linguistics, Stroudsburg (2011). http://dl.acm.org/citation.cfm?id=2002472.2002642
44. Shen, W., Wang, J., Luo, P., Wang, M.: Linking named entities in tweets with knowledge base via user interest modeling. In: Dhillon, I.S., Koren, Y., Ghani, R., Senator, T.E., Bradley, P., Parekh, R., He, J., Grossman, R.L., Uthurusamy, R. (eds.) The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, 11–14 August 2013, pp. 68–76. Association for Computing Machinery, New York (2013). http://dl.acm.org/citation.cfm?id=2487575
45. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web, WWW '07, pp. 697–706. Association for Computing Machinery, New York (2007). http://doi.acm.org/10.1145/1242572.1242667
46. Varma, V., Bysani, P., Reddy, K., Reddy, V.B., Kovelamudi, S., Vaddepally, S.R., Nanduri, R., Kumar, N.K., Gsk, S., Pingali, P.: IIIT Hyderabad in guided summarization and knowledge base population. In: TAC. NIST (2010). http://www.nist.gov/tac/publications/2010/papers.html
47. Welty, C., Murdock, J.W., Kalyanpur, A., Fan, J.: A comparison of hard filters and soft evidence for answer typing in Watson. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) The Semantic Web – ISWC 2012, pp. 243–256. Springer, Berlin (2012)
48. West, R., Gabrilovich, E., Murphy, K., Sun, S., Gupta, R., Lin, D.: Knowledge base completion via search-based question answering. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 515–526. Association for Computing Machinery, New York (2014)
49. Zhang, W., Su, J., Tan, C.L., Wang, W.: Entity linking leveraging automatically generated annotation. In: Huang, C.R., Jurafsky, D. (eds.) COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings of the Conference, 23–27 August 2010, Beijing, pp. 1290–1298. Tsinghua University Press (2010). http://aclweb.org/anthology/C/C10/
50. Zheng, Z., Li, F., Huang, M., Zhu, X.: Learning to link entities with knowledge base. In: HLT-NAACL, pp. 483–491. The Association for Computational Linguistics (2010). http://www.aclweb.org/anthology/N10-1072

Chapter 9
Clustering Multi-View Data Using Non-negative Matrix Factorization and Manifold Learning for Effective Understanding: A Survey Paper

Khanh Luong and Richi Nayak

Abstract Multi-view data that contains the data represented in many types of features has received much attention recently. The class of method utilizing non-negative matrix factorization (NMF) and manifold learning to seek the meaningful latent structure of data has been popularly used for both traditional data and multi-view data. The NMF and manifold-based multi-view clustering methods focus on dealing with the challenges of manifold learning and applying manifold learning on the NMF framework. This paper provides a comprehensive review of this important class of methods on multi-view data. We conduct an extensive experiment on several datasets and raise many open problems that can be dealt with in the future so a higher clustering performance can be achieved.

9.1 Introduction

With the rapid growth of computing technology, datasets in real world are getting richer in terms of both semantic and structures. This has led to represent the data from multiple perspectives and has given rise to a new term "multi-view" for this type of data. Data analytic methods that exploit multi-view data result in generating more meaningful and realistic outcomes by bringing the complementary and compatible information together [1]. The multi-view clustering problem can be seen as a process of simultaneously learning from multiple views to yield the consensus and general grouping information of the dataset.

Due to the data representing with multiple views, the underlying data model is usually very sparse and high dimensional. Finding clusters on the original search space is very expensive and results in an unrealistic and unstable clustering solution.

K. Luong (✉) · R. Nayak
Queensland University of Technology, Brisbane, QLD, Australia
e-mail: khanh.luong@hdr.qut.edu.au; r.nayak@qut.edu.au

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_9

Therefore most of the existing multi-view learning methods attempt to learn the low-dimensional latent features of data as a first step for further processing. These methods are based on subspace learning or non-negative matrix factorization (NMF) framework [1–4]. Between these two approaches, the multi-view clustering methods that utilize NMF have been received much attention and proved to be more effective due to their ability to learn the effective latent features by projecting original high-dimensional to a low-dimensional space [5, 6]. However, the NMF framework respects the Euclidean space only and fails to preserve the geometric structures of original data when projecting it to lower order [7]. In other words, the neighbouring objects in original space may not remain neighbours in the new space. Maintaining of the geometric structure in lower order is necessary for learning a meaningful representation and effective understanding of data and clusters.

Manifold learning [8] is a newly emerging approach in dimensionality reduction aiming at preserving the local and/or the global geometric structures of the original data [9, 10]. Manifold learning is combined with the NMF framework to discover and maintain the geometric structure of data when projecting it to a lower dimensional embedded space [1, 4, 11, 12]. Clustering methods based on NMF with the aid of manifold learning have shown the effectiveness over the other state-of-the-art methods. Unfortunately, it is not a trivial task to find the intrinsic manifold of the data. Finding the inaccurate manifold can cause poor clustering outcome. How to achieve the accurate manifold of the data is hard for the traditional NMF-based clustering methods. This challenge is increased significantly when applying NMF and manifold learning on a multi-view clustering problem.

Moreover, recent research has pointed out that though data is represented with multiple views, the manifolds of data are believed to be lying on a convex hull of the manifold of all views [1]. Hence, when using manifold learning on multi-view, in addition to the difficulty of learning the accurate manifold on each view, multi-view learning should also deal with how to learn the accurate consensus manifold from multiple views and then learn the meaningful consensus coefficient matrix respecting the consensus manifold. It has been evidenced in the literature that it is not a trivial task to effectively employ manifold learning for NMF-based multi-view data for effective understanding. Researchers have proposed a number of methods to deal with this problem ranging from focusing on learning the accurate manifold on each data view [4, 11, 12] to learning the correct consensus manifold from multiple views after having manifold learned on each view [1]. This is an emerging research field that needs attention and requires to be presented in a systematic fashion. There has been no comprehensive discussion of this specific field of multi-view clustering and associated methods. We propose to present a survey paper that will (1) provide a general view of how to understand multi-view problem via applying NMF and learning manifold; (2) include an experimental study on well-known multi-view datasets to investigate the effectiveness of the methods on datasets and (3) discuss the challenging problems that can be addressed in the future research. To the best of our knowledge, this will be the first survey paper to summarize a class of clustering methods combining NMF and manifold learning to effectively understand the challenging multi-view data problem.

9.2 Multi-View Data Clustering

9.2.1 Overview of Multi-View Data

With the availability of collecting data from different sources or extracted from different feature extractors, the multi-view data is becoming ubiquitous. For example, the image data can naturally be described by different aspects of visual characteristics [1]. The multiple views in this data can be generated by the processes such as scale-invariant feature transform (SIFT) [13], speeded-up robust features (SURF), histogram of oriented gradients (HOG) [14], local binary patterns (LBP) [15], GIST [16], and CENTRIST [17]. A web-based dataset can be represented by multiple views derived by content, hyperlink (inbound-link or outbound-link) and usage logs. In a multilingual dataset, multiple views are derived by different languages used for representing original documents. A multimedia data can be represented by multiple views of colour, texture, shape, etc., and a bioinformatics dataset can be represented with multiple views of mRNAexpression, RNA sequence and protein–protein interaction. The multi-view data not only contains valuable, compatible and complementary information but also contains a diversity of different views. Figure 9.1 shows an example of multi-view data.

A special but popular case of multi-view data occurs when data is missing in some views, named as incomplete or partial multi-view data [18–20]. An example is a multilingual dataset where all documents do not have equivalent translation

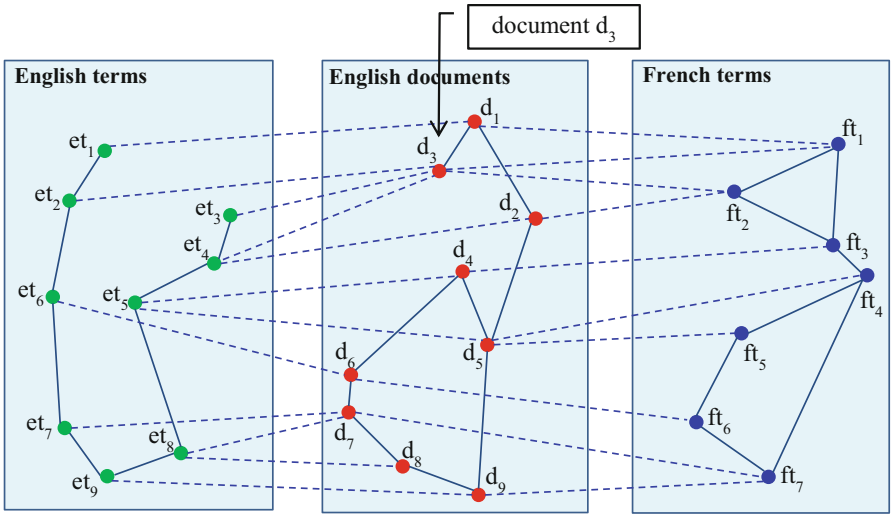


Fig. 9.1 An example of multi-view/multi-type relational data. The multi-view dataset is represented by two views: English terms and French terms. The MTRD dataset has three object types: English documents, English terms and French terms. The intra-type relationships are represented as solid lines and inter-type relationships are represented as dotted lines

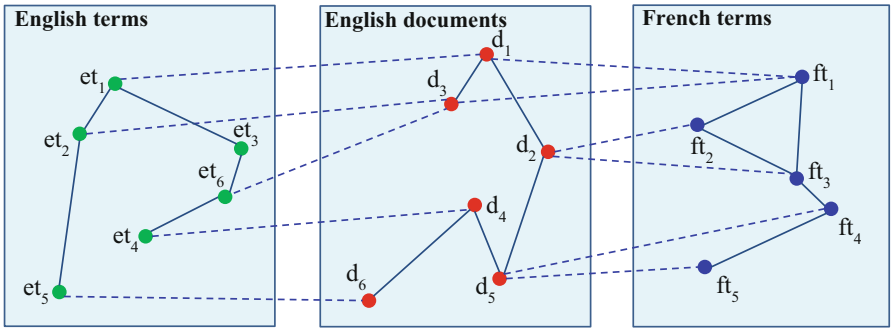


Fig. 9.2 An example of partial multi-view dataset

on every language. As illustrated in Fig. 9.2, documents d_2 and d_5 do not have representation on view 1 (English terms) and documents d_4 and d_6 do not have corresponding representations on view 2 (French terms). Another class of multi-view data is multi-type relational data (MTRD) where a sample object is represented with multiple relationships. An MTRD dataset will include different types of objects as well as different types of relationships including inter-type and intra-type. An intra-type relationship models the relationships between objects of the same type and an inter-type relationship describes the relationships between objects of two different types. Studying a sample object in relationships with other type objects is similar to considering the object in different views. A multi-view dataset can be considered as an MTRD dataset. For example, the multi-view multilingual dataset mentioned above is also an MTRD. In this case, the document object type is the sample object type and different object types corresponding to different languages are different feature object types, as shown in Fig. 9.1. Under multi-view data, documents in the dataset are represented under view 1 (English terms) or view 2 (French terms). On view 1, document d_3 will be represented by three English terms et_2, et_3 and et_4. On view 2, i.e., when looking at the French translation version of all documents, the document d_3 will be represented by two French terms ft_1 and ft_2. The representations of samples on features of different views are plotted as dashed lines and the similarities between samples or between features on each view are represented as solid lines in Fig. 9.1. Under multi-type relational data setting, documents in the dataset can have multiple relations with other object types, i.e., English terms and French terms. For example, document d_3 has relationships with three English terms et_2, et_3 and et_4 and at the same time d_3 has relationships with French terms ft_1 and ft_2. Inter-type relationships between sample objects and different feature objects are represented by dashed lines and the intra-type relationships between objects of the same type are represented by solid lines in Fig. 9.1.

9.2.2 NMF-Based Multi-View Data Clustering Definition

Suppose $X = \{X_1, X_2, \ldots, X_{n_v}\}$ is a multi-view dataset with n_v views in total and the data in view vth is represented in the data matrix $X_v \in R_+^{n \times m_v}$, where n is the number of samples and m_v is the number of features of the vth view. The multi-view data NMF-based clustering task is to group data samples into clusters by using the low-rank space learned by utilizing all complementary and compatible views data in the factorizing process.

There are two popular forms of objective function for the NMF-based multi-view low-dimensional learning. The first form follows the objective function introduced in MultiNMF [21] and used in many other works [19, 22–24], written as below,

$$\min \sum_{v=1}^{n_v} \|X_v - H_v W_v\|_F^2, \text{ s.t. } H_v \geq 0, W_v \geq 0 \tag{9.1}$$

where $H_v \in R_+^{n \times r}$ is the new low-rank representation of data corresponding to the basis $W_v \in R_+^{r \times m_v}$ under the vth view, r denotes the number of the new rank. The optimizing process in this objective function, similar to conventional NMF objective function [5], is to update $\{H_v\}_{v=1\ldots n_v}$ and $\{W_v\}_{v=1\ldots n_v}$, returning in the optimal low-rank matrices H_v and W_v such that $H_v W_v$ is a good approximation of X_v for all $v = 1 \ldots n_v$.

In the fusion step, the consensus latent feature matrix of all views, denoted as H_*, is normally calculated by linearly combining all individual low-rank representations as in [22],

$$H_* = \sum_{v=1}^{n_v} H_v / n_v \tag{9.2}$$

or taking the average as in [23],

$$H_* = [H_1 \ldots H_{n_v}] \tag{9.3}$$

or effectively learning at the same time as the low-rank representations are learned via the factorizing step as in [21],

$$\min \sum_{v=1}^{n_v} \|X_v - H_v W_v\|_F^2 + \sum_{v=1}^{n_v} \|H_v - H_*\|_F^2, \text{ s.t. } H_v \geq 0, H_* \geq 0, W_v \geq 0 \tag{9.4}$$

The objective function in Eqs. (9.1)–(9.4) is able to simultaneously learn different low-rank data representations from different data views. In the later step, the consensus data matrix will be learned via compensating the newly learned data representations from all views.

The second form to learn the low-dimensional representations for multi-view data is utilized in multi-view coupled matrix factorization (MVCMF) [25], inspired from the data fusion topic [26, 27],

$$\min \sum_{v=1}^{n_v} \|X_v - H_* W_v\|_F^2, \text{ s.t. } H_* \geq 0, W_v \geq 0 \tag{9.5}$$

where $W_v \in R_+^{r \times m_v}$ is the basis representation of data under the vth view and H_* is the consensus coefficient matrix learned from all views. In this objective function, factor matrices to be updated including H_* and $\{W_v\}_{v=1...n_v}$. Using this objective function, the consensus coefficient matrix will be learned directly during the learning process. This can help reducing the computation cost since a fusion step to learn the consensus matrix will be unnecessary.

For MTRD setting, apart from sample object type, different feature views can be considered as different feature object types. The data represented on each view is corresponding to the inter-type relationship between sample object type and each feature object type. In particular, data view X_v is the inter-type relationship between sample and feature object type vth. We use R_{hl} to denote the inter-type relationship between object type hth and object type lth. The number of inter-relationship matrices in MTRD setting may be larger than the number of data matrices in multi-view setting since there may exist relationships between different feature object types. For example, a web-page dataset with two-view data, i.e., between web-pages and terms and between web-pages and concepts may have three inter-type relationship matrices such as between web-pages and terms, between web-pages and concepts and between terms and concepts. The objective function for the MTRD low-rank learning normally utilizes the non-negative matrix tri-factorization (NMTF) framework [28]. It can be expressed as,

$$\min \sum_{h,l=1}^{m} \|R_{hl} - G_h S_{hl} G_l^T\|_F^2, \text{ s.t. } G_h \geq 0, G_l \geq 0 \tag{9.6}$$

where m is the number of object types, $G_h \in R_+^{n_h \times r}$, $G_l \in R_+^{n_l \times r}$ and $S_{hl} \in R_+^{r \times r}$. G_h and G_l are the low-rank factor matrices of object types hth and lth. Learning the cluster structure of object type hth will enhance the clustering performance on object type lth while the clustering process on object type hth can be determined by the clustering of object type lth at the same time. Consequentially, simultaneously learning cluster structures of all object types will help enhancing the clustering performance of the sample object type. With regard to the equivalence between the MTRD and multi-view setting, let us suppose object type h is the sample object type and object type l is the feature view l, it can be inferred that R_{hl}, G_h and G_l in Eq. (9.6) are equivalent to X_l, H_l and W_l in Eq. (9.1), respectively.

The problem of simultaneous clustering of multi-type relational data in Eq. (9.6) can be complicated in solving. Therefore, a variation using symmetric non-negative matrix tri-factorization (STNMF) objective function has been proposed in [29] and

has also been used in other methods [11, 12]. In the STNMF setting, all the inter-type relationships between different object types are encoded using the new formulation of matrix R in Eq. (9.7). Note that R is a symmetric matrix since $R_{hl} = R_{lh}^T$.

$$R = \begin{bmatrix} 0^{n_1 \times n_1} & R_{12}^{n_1 \times n_2} & \cdots & R_{1m}^{n_1 \times n_m} \\ R_{21}^{n_2 \times n_1} & 0 & \cdots & R_{2m}^{n_2 \times n_m} \\ \vdots & \vdots & \ddots & \vdots \\ R_{m1}^{n_m \times n_1} & R_{m2}^{n_m \times n_2} & \cdots & 0^{n_m \times n_m} \end{bmatrix} \tag{9.7}$$

The STNMF objective function is written as

$$\min \left\| R - GSG^T \right\|_F^2, \text{ s.t. } G \geq 0 \tag{9.8}$$

In this objective function, S is the trade-off matrix and has the symmetric form similar to R. G is the symmetric factor matrix that contains all low-rank factor matrices of all object types in the following form:

$$G = \begin{bmatrix} G_1^{n_1 \times r} & 0^{n_1 \times r} & \cdots & 0^{n_1 \times r} \\ 0^{n_2 \times r} & G_2^{n_2 \times r} & \cdots & 0^{n_2 \times r} \\ \vdots & \vdots & \ddots & \vdots \\ 0^{n_m \times r} & 0^{n_m \times r} & \cdots & G_m^{n_m \times r} \end{bmatrix} \tag{9.9}$$

where m is the number of object types, equal to the number of views plus 1. The symmetric formulation in Eq. (9.8) can learn factor matrices for all object types simultaneously since all the inter-relationship information have been utilized at the same time during the learning process. The objective function in Eq. (9.8) has provided an elegant and flexible formula for the problem of representing the MTRD. This formulation of objective function can be effectively extended to incorporate other regularizations in order to obtain the higher clustering performance; however, as per our experiments, the methods based on symmetric NMF require more running time as compared to the other methods based on the objective function in Eq. (9.6).

9.3 NMF Framework and Manifold Learning on Traditional Data

NMF is an established method to find an approximate product of smaller matrices equivalent to the original data matrix for learning lower rank matrices. In clustering, an NMF-based method attempts to project data into the new lower dimensional space and then seeks the latent groups in the new low embedded space. Due to the ability to learn part-based representations, NMF has been proved to be the effective and flexible clustering method, especially in very high-dimensional data [30]. In the first NMF-based clustering method [30], NMF derived the latent semantic space

and determined the document groups based on how each document is combined with each document topic cluster. This first method deploying NMF to document clustering marked a new era. Another NMF based typical work on traditional one view data is ONMTF (orthogonal non-negative matrix factorization) [28]. This work applied the orthogonal constraint on factor matrices of both samples and features, leading to both non-negative and orthogonal constraints are used at the same time. These two constraints on factor matrices turn these matrices into cluster indicator matrices, thus make ONMTF be a well-defined NMF co-clustering-based method. This establishes that the NMF-based clustering solution is not only unique but it is also meaningful and interpretable. NMF-based clustering methods have shown the effectiveness of NMF over other traditional clustering methods such as K-means or hierarchical clustering methods. However, NMF-based methods that focus on finding the approximate factor matrices have been criticized to fail to preserve the geometric structure of data [7]. In other words, the geometric structure of the learned low-order representation may not share the similar geometric structure as the original structure of the original data.

On the other hand, the manifold learning [8] algorithms have been known to learn the intrinsic manifold or geometric structure of data. These algorithms work on the assumption that the nearby points should have similar embeddings. This assumption requires the learning process to ensure the distances between points remain unchanged.

9.3.1 Formulation of Manifold Learning on Traditional Data

The formulation of manifold learning has been addressed in many previous works [8, 31], we make a review here for the benefit of the paper's logicality.

Suppose H is the low-rank latent feature matrix on the new mapped space learnt from data matrix X. We have the following optimizing term to ensure the data points in new space are smooth with the intrinsic geometric structure of the original data.

$$\min \sum_{i,j=1}^{n} \|h_i - h_j\|^2 a_{ij} \tag{9.10}$$

where $A = \{a_{ij}\}^{n \times n}$ is the adjacency matrix captured by building the k nearest neighbour (kNN) graph, a_{ij} is defined as [7, 8],

$$a(i, j) = \begin{cases} t_{ij} \text{ if } x_i \in \mathcal{N}_k(x_j) \text{ or } x_j \in \mathcal{N}_k(x_i) \\ 0, \text{ otherwise} \end{cases} \tag{9.11}$$

where $T = \{t_{ij}\}$ represents the input affinity information between all pairs of data points x_i and x_j, e.g., the similarities between pairs of documents in a text dataset. $\mathcal{N}^k(x_i)$ denotes k nearest neighbours of x_i. It can be obviously inferred from the

definition of A that the adjacency matrix A contains only the high relatedness values of data points that are residing in the k nearest neighbour areas. If two data points x_i and x_j are not in neighbourhood area of each other, their corresponding value a_{ij} will be set to be zero. $\|h_i - h_j\|^2$ in Eq. (9.10) is the Euclidean distance estimating the closeness between two new representations h_i and h_j projected for arbitrary x_i, x_j data points.

There are two main cases for data points when they are projected to the new lower dimensional space.

Case 1 If two data points x_i, x_j are far in the original space, i.e., $a_{ij} = 0$, the optimizing term in Eq. (9.10) will not bring any effect to the learning process.

Case 2 If two data points x_i, x_j are close in the original space, i.e., $a_{ij} > 0$, their new representations h_i, h_j should share the same neighbourhood or their distance $\|h_i - h_j\|^2$ should be as small as possible. The optimizing term in Eq. (9.10) will minimize the distance between two representations h_i and h_j when $a_{ij} > 0$ (i.e., x_i and x_j are close in the original data space). Eq. (9.10) can be equivalently written as below,

$$\Leftrightarrow \min \left(\sum_{i=1}^{n} h_i h_i^{'} \sum_{j=1}^{n} a_{ij} + \sum_{j=1}^{n} h_j h_j^{'} \sum_{i=1}^{n} a_{ij} - 2 \sum_{i=1}^{n} \sum_{j=1}^{n} h_i h_j^{'} a_{ij} \right) \tag{9.12}$$

$$\Leftrightarrow \min \left(2Tr\left(H^T DH\right) - 2Tr\left(H^T AH\right) \right) \tag{9.13}$$

$$\Leftrightarrow \min Tr\left(H^T LH\right) \tag{9.14}$$

where $Tr(.)$ denotes the trace of a matrix. D is the diagonal matrix where $d_{ii} = \sum_{j=1}^{n} a_{ij}$ and $L = D - A$ is the Laplacian graph. The minimizing term in Eq. (9.14) is called the manifold learning regularization [7, 8]. This manifold learning is combined with NMF framework in clustering to find accurate and meaningful clustering solutions while maintaining the original structure of data. Since NMF factorizes the higher-order data matrix into smaller matrices, it applies many approximations. Incorporating manifold learning into NMF framework helps approximating the factor matrices that maintain the geometric structure of the original data and produces meaningful clusters. On the traditional data, GNMF [7] was the first method to incorporate manifold learning into NMF framework to learn the low-rank representations that preserve the local geometric structure. Due to maintaining distances between neighbouring points when projecting to lower order, close points in original space are guaranteed to share the same cluster. This helps resulting in the meaningful cluster solution. The objective function of GNMF on traditional one-view data is defined as below:

$$\min \|X - HW\|_F^2 + Tr\left(H^T LH\right), \text{ s.t. } H \geq 0, W \geq 0 \tag{9.15}$$

where L is the Laplacian graph, constructed as in Eq. (9.14).

9.4 NMF Framework and Manifold Learning on Multi-View Data

On multi-view data, the NMF objective function to learn the latent features with the local geometric structure preserved on all views is as [1, 20],

$$\min \sum_{v=1}^{n_v} \left(\|X_v - H_v W_v\|_F^2 + Tr\left(H_v^T L_v H_v\right) \right), \text{s.t. } H_v \geq 0, W_v \geq 0 \tag{9.16}$$

where $L_v = D_v - A_v$ is the Laplacian graph and A_v is the adjacency matrix built on data view vth. The second term in this objective function helps the learning process to return in the low-rank latent feature matrices $\{H_v\}$ that are smooth with the intrinsic geometric structure of data view vth and thus can be more meaningful as compared to latent feature matrices $\{H_v\}$ obtained from the multi-view NMF objective function in Eq. (9.1).

Applying manifold learning on NMF-based clustering methods for multi-view data is not a trivial extension of traditional data. Since data on different views may sample on different structures, there are three main family of methods according to the way they deal with this challenge.

Firstly, a natural extension of manifold learning on multi-view data is learning the intrinsic manifold of data on each view and ensuring the low-rank data in the new mapped space is smooth with the intrinsic manifold on the corresponding view. This first family of methods focus on learning and preserving the manifold on each view to learn different view factor matrices and constrain the consensus matrix to be the best consensus of all factor matrices.

Secondly, similar to the case of using manifold on traditional data, the most important thing is to learn the accurate manifold embedded on each view and constrain the low-rank representation on each view to lie on the corresponding intrinsic manifold. This will help to achieve accurate cluster structures from each view, the necessity to learn the correct consensus coefficient matrix. This second family of methods rely on learning the accurate manifold on each view to learn the meaningful common matrix.

Thirdly, some state-of-the-art methods regard that the intrinsic manifold in the dataset is embedded in a convex hull of all the manifolds of all views [1, 12]. The convex hull of all manifolds from all views can be the minimal convex set containing all manifolds or a convex combinations of all manifolds [32]. After learning the consensus manifold, it is a vital problem to learn the low-order representations to ensure the smoothness with the intrinsic manifold learned from all views. Therefore, this emerging family of methods focus on learning the consensus manifold embedded from multiple manifolds from all views. They rely on the learned consensus manifold to find the consensus coefficient matrix.

This section provides a comprehensive review of the three family of NMF-based clustering methods on multi-view data.

9.4.1 Learning the Consensus Matrix Relying on Learning the Manifold on Each View

An easy extension of transforming the application of manifold learning from a single view to multi-view is learning manifold of data on each view and attempting to preserve the manifold when learning the low-dimension data on each view.

Based on the assumption that different views data should embed similar cluster structures, authors in [24] proposed the objective function that attempts to learn the low-rank representations from all views such as these representations are as similar as possible. At the end of the optimizing process, it is believed that each coefficient matrix will become the consensus coefficient matrix and will embed the cluster structure. In order to ensure the meaningfulness of the consensus matrix, the graph regularization is incorporated into the objective function to help to learn the low-rank representations respecting their corresponding manifolds. The objective function is as

$$\min \sum_{v=1}^{n_v} \left(\|X_v - H_v W_v\|_F^2 + \sum_{v=1}^{n_v} \sum_{s=1}^{n_v} \|H_v - H_s\|_F^2 + \alpha R + \lambda \sum_{v=1}^{n_v} Tr\left(H_v^T L_v H_v\right) \right),$$
$$\text{s.t. } W_v \geq 0, H_v \geq 0 \tag{9.17}$$

The first term in the objective function requires H_v to be the best low-rank representation learned from data view X_v. The second term requires that coefficient matrix H_v must be similar to other coefficient matrices of other views. The third term $R = \min \sum_{v=1}^{n_v} \|(H_v)^T - I\|_F^2$ is added to emphasize the orthogonality constraint on factor matrix H_v. The advantage of the constraint is to returning the unique and interpretable matrix [28]. The last term is to guarantee the manifold assumption of every data point on the low-dimensional space, i.e., the low-rank data points should lie on the embedded manifold of the corresponding view.

It can be noted that the low-rank representation H_v from each view is restricted by using many constraints in objective function Eq. (9.17). This helps returning in the good low-order representations on the well-defined datasets, yet it can cause bad results on some datasets because of too many constraints.

In the context of MTRD, dual regularized co-clustering (DRCC) [33] is designed for co-clustering data, i.e., simultaneously clustering samples and features in a dataset where data samples are represented by one type feature. Co-clustering or clustering on bi-type data is based on the duality between samples and feature and is a special case of multi-view (i.e., one-view data) clustering and MTRD (i.e., two-type data) clustering. DRCC is the first method designed for manifolds on both data samples and data features though the co-clustering problem was first presented in [34]. In DRCC objective function, the sample and feature are simultaneously being clustered. This co-clustering process will be able to investigate the structures of features and the clustering of features will boost the clustering process of samples.

The more accurate feature clusters are learned, the more accurate sample clusters will be achieved. Therefore simultaneously seeking the low-rank representations of samples and features as well as simultaneously maintaining the geometric structures of low-dimensional spaces of both samples and features as in DRCC objective function is believed to be the most effective learning process for the co-clustering problem.

In DRCC, two graphs, one for data samples and the other for data features, are constructed to model the local geometric structure information. This graph regularization is incorporated in the NMF objective function to introduce the smoothness of both data points and data features with their embedded manifolds. The objective function is as below:

$$\begin{aligned} &\min\left(\left\|X - HSF^T\right\|_F^2 + \lambda Tr\left(H^T L_H H\right)\right) + \mu Tr\left(F^T L_F F\right)\right) \\ &\quad \text{s.t. } H \geq 0, F \geq 0 \end{aligned} \tag{9.18}$$

It is noted in DRCC that the l_2-norm constraint is assigned on rows of H and F to ensure the objective function is lower bounded and the trade-off matrix S is relaxed to take any signs.

As a formal extension of DRCC, STNMF [29] proposed to learn manifolds for all object types in an MTRD dataset. The objective function is as

$$\min \left\|R - GSG^T\right\|_F^2 + \lambda Tr\left(G^T LG\right), \text{ s.t. } G \geq 0 \tag{9.19}$$

where the Laplacian L is symmetrically defined as

$$L = \begin{bmatrix} L_1^{n_1 \times n_1} & 0^{n_1 \times n_2} & \cdots & 0^{n_1 \times n_m} \\ 0^{n_2 \times n_1} & L_2^{n_2 \times n_2} & \cdots & 0^{n_m \times n_2} \\ \vdots & \vdots & \ddots & \vdots \\ 0^{n_m \times n_1} & 0^{n_m \times n_2} & \cdots & L_k^{n_m \times n_m} \end{bmatrix} \tag{9.20}$$

L_1 is the Laplacian defined on sample object type, L_2 is the Laplacian defined on feature object type 1 and so on. The symmetric inter-type relationship R and the symmetric factor matrices G are defined as in Eqs. (9.7) and (9.9), respectively. STNMF was the first method utilizing NMF to simultaneously cluster many object types. STNMF proposed a novel symmetric framework for the MTRD clustering problem and incorporated Laplacian matrices to learn manifolds on sample object types and all feature object types.

For the context of partial multi-view data, inspired from the NMF-based method for partial data [18], DCNMF [20] was designed using the NMF framework, but incorporates manifold learning while looking for the meaningful common coefficient matrix. On the one hand, the method learns the manifold of data on every view and aims to preserve the manifold as a constraint of the objective function to be followed. Similar to the other methods [1, 24], the low-rank representations learned

from all views will be ensured to be smooth with the intrinsic shape of each view. On the other hand, based on the assumption of the cluster similarity constraint, the objective function is designed to ensure the paired examples in the partial multi-view data to be as similar as possible. For this goal, the coefficient matrices of different views are forced to be learned toward the common consensus matrix. By this way, the common matrix is learned during the optimization process and at the same time with views' coefficient matrices.

Similar to [20], the graph regularized partial multi-view clustering (GPMVC) method [19] attempts to learn the meaningful representation on each view by respecting the geometrical structure of data on each view when using NMF to project higher-dimensional to lower-dimensional space. The objective function is as below:

$$\min \sum_{v=1}^{n_v} \left(\|X_v - H_v W_v\|_F^2 + \mu_v \|H_v Q_v - H_*\|_F^2 + \lambda_v Tr\left(H_v^T L_v H_v\right) \right) \tag{9.21}$$

where Q_v is a diagonal matrix, constructed as $(q_v)_{jj} = \sum_i (w_v)_{ij}$. Q_v is used to normalize the coefficient matrix H_v to help different H_v from different views are scale comparable. With the objective function, H_* is believed to contain meaningful latent features learned from all views as the learning process maintains the closeness between points in the neighbourhood area on each view. Though these methods [19, 20] provide an effective approach for partial multi-view clustering, due to avoiding the learning of consensus manifold, the low-rank representations may not reflect the true geometric structure of original data.

9.4.2 *Learning the Consensus Coefficient Matrix Relying on Learning the Accurate Manifold on Each View*

Manifold learning can aim at preserving the local or global geometric structures of data [10]. Since locality preserving shows a tendency to group similar data, most manifold learning methods incorporated in clustering aim at preserving the local structure of the data. These manifold learning methods rely on building a kNN graph, a powerful technique used widely in machine learning and data mining. The kNN graph models the neighbourhood information for each data point and encodes the closeness between each data point to all its neighbours. This closeness information will be embedded in the learning process to make sure the projection respect the local geometric structure of data. Since the closeness of data objects can be varied depending on different types of data, the pairwise similarity can be calculated based on different similarity weighted schemes such as Euclidean distance, Cosine similarity or Gaussian kernel.

Relational multi-manifold co-clustering (RMC) [12] was proposed in an effort to learn the accurate intrinsic manifold for MTRD data. RMC learns the optimal

manifold from a combination of many predefined manifolds. The predefined manifolds are built as initial guesses of graph Laplacian. Different guesses can be constructed by different kinds of weighted schemes or different values for the neighbourhood size parameter. In RMC, the intrinsic manifold is calculated by the following equation:

$$L = \sum_{i=1}^{q} \mu_i \widetilde{L}_i, \text{ s.t.} \sum_{i=1}^{q} \mu_i = 1, \mu_i \geq 0 \tag{9.22}$$

where $\widetilde{L}_i$ denotes a candidate manifold i and q is the number of candidate manifolds. Each candidate manifold is a combination of different manifolds of different object types expressed as the symmetric form equation,

$$\widetilde{L}_i = \begin{bmatrix} L_1 & 0 & 0 \\ 0 & L_2 & 0 \\ \cdots & \cdots & \cdots \\ 0 & 0 & L_m \end{bmatrix} \tag{9.23}$$

L_1 is the Laplacian of samples data and different L_i are different manifolds of different feature object types. The candidate Laplacian graph $\widetilde{L}_i$ will be embedded in the symmetric framework with the objective function below:

$$J = \min \left\| R - GSG^T \right\|_F^2 + \alpha Tr \left(G^T \left(\sum_{i=1}^{q} \mu_i \widetilde{L}_i \right) G \right) + \beta \|\mu\|^2 \tag{9.24}$$

R and G are non-negative and formulated as in Eqs. (9.7) and (9.9), respectively. l_2-norm on μ is for the even distribution of the parameter for all manifolds and it prevents the parameter overfitting to one manifold. Since RMC considers several possible manifolds, the learned consensus manifold is believed to be the closest to the intrinsic manifold of original data. Though it learns many manifolds but they are of the same type, i.e., based on kNN graph. Consequently, the learned manifold in RMC is less diverse while requiring the extra computational cost to calculate the ensemble manifold. In addition, the parameter k for building kNN graph can not be known a priori. This makes the use of kNN an uncertainty. Knowing the right value for neighbourhood size k helps to choose all useful neighbours when constructing the kNN graph. Thus it helps to learn a more meaningful affinity matrix that is a necessity to learn the accurate manifold and achieving meaningful clustering solution. However, a big value of k may lead to involving non-useful neighbour points in the kNN graph and can lead to learning an inaccurate manifold and a small value of k may not include all useful neighbour points and will lead to learning an incomplete manifold. To ensure learning a more diverse manifold or ensure all useful neighbour information is included in constructing the affinity

matrix, RHCHME proposed to build the manifold by combining Euclidean distance learning and subspace learning. The complete manifold is learned as

$$L = \alpha L^E + L^S \tag{9.25}$$

where $L^S = D - A^S$ with D is the diagonal matrix where $(d)_{ii} = \sum_j (a^S)_{ij}$, A^S is the similarity matrix learned from considering data points lying on subspaces, i.e., two data points will be considered as neighbours if they belong to the same subspace, despite the distance between them [35]. $L^E = D - A^E$ with $(d)_{ii} = \sum_j (a^E)_{ij}$, A^E is the affinity matrix derived from constructing kNN graph as in Eq. (9.11).

The Laplacian matrix constructed as in RHCMHE can learn a more comprehensive intrinsic manifold for data when projecting. The objective function of RHCHME is based on STNMF framework and is similar to RCM. However, the Laplacian graph is not an ensemble of many candidate manifolds but is built by considering the data lying on manifold as well as belonging to subspace. The more comprehensive manifold learned in RHCHME helps to provide more meaningful clusters evidenced by more accurate clustering results. However, similar to other clustering methods using manifold learning, RHCHME learns the manifold by considering local geometric structure only. The local geometric-based learning manifold aims at preserving the close distances between neighbour points and/or ensuring membership to subspace of data points. RHCHME is similar to most manifold learning algorithms that avoid considering distance information of data points that do not share the same neighbourhood due to the computation cost and the local distance has been proved to be more useful to clustering than the global distance. However, it has been shown that preserving all distance information both for close and far points will help to learn a more meaningful representation [10]. Inspired by this, ARASP [4] learns an accurate manifold where all important distance information is captured in an effective manner. Similar to RHCHME and RMC, ARASP is designed on the setting of MTRD. ARASP proposes to build the affinity matrix in a novel manner utilizing k nearest neighbour (kNN) graph for close distances and p farthest neighbour (pFN) graph for far distances. The newly affinity matrix is built ready to be regularized and embedded in the objective function to ensure maintaining all important distances during the manifold learning process. The affinity matrix on data type X_h is constructed as

$$A_h = \lambda A_h^n - \beta A_h^f, \lambda + \beta = 1 \tag{9.26}$$

where A_h^n is the weighted adjacency matrix, built from constructing kNN graph, with the goal of preserving the local geometric structure of data. A_h^n is constructed as in Eq. (9.11). A_h^f is the weighted repulsive matrix, built with the goal of keeping dissimilar points far apart in the new mapping space. A_h constructed as in Eq. (9.26) allows including both close and far distances of original data, therefore, can help

resulting in a more complete manifold for learning process. $A_h^f = \{a_h^f(i,j)\}^{n_h \times n_h}$ is defined as

$$a_h^f(i,j) = \begin{cases} t_{ij} \text{ if } x_i \in \mathscr{F}_p(x_j) \text{ or } x_j \in \mathscr{F}_p(x_i) \\ 0, \text{ otherwise} \end{cases} \tag{9.27}$$

where $\mathscr{F}_p(x_i)$ denotes the p farthest neighbour points of x_i. The ARASP objective function is as

$$\begin{aligned} J_1 &= \min \left(\sum_{1 \le h < l \le m} \left\| R_{hl} - G_h S_{hl} G_l^T \right\|_F^2 - \sum_{1 \le h \le m} Tr\left(G_h^T L_h G_h\right) \right), \\ &\text{s.t. } G_h \ge 0, G_l \ge 0, S_{hl} \ge 0 \end{aligned} \tag{9.28}$$

$L_h = D_h - A_h$ is the Laplacian graph defined on object type hth with the affinity matrix A_h defined as in Eq. (9.26). As pointed out in [4], the pseudo orthogonal constraint (i.e., the normalize cut-type constraint [36]) is applied on factor matrix such that $G_h^T D_h G_h = I, \forall h = 1 \ldots m$, the ARASP objective function becomes

$$\begin{aligned} J_1 &= \min \left(\sum_{1 \le h < l \le m} \left\| R_{hl} - G_h S_{hl} G_l^T \right\|_F^2 - \sum_{1 \le h \le m} Tr\left(G_h^T A_h G_h\right) \right), \\ &\text{s.t. } G_h \ge 0, G_h^T D_h G_h = I, G_h 1_r = 1_{n_h}, \forall h = 1 \ldots m, \\ &S_{hl} \ge 0, S_{hl} 1_r = 1_r, 1 \le h < l \le m \end{aligned} \tag{9.29}$$

The constraints $G_h 1_r = 1_{n_h}$ and $S_{hl} 1_r = 1_r$ are to apply $l1$-normalization on each row of the factor matrix G_h and the association relationship matrix S_{hl}, respectively. The normalize cut-type constraint on factor matrix has forced the trade-off matrix S_{hl} to become the association relationship between clusters of different object types. This will help the corresponding factor matrices to approach the optimal solution. Furthermore, ARASP does not rely on using the big symmetric matrix as other MTRD methods do but directly using the original NMF framework. Working with the smaller matrices significantly improves the computational complexity and results in time saving as evidenced by empirical analysis.

Though the above methods show effectiveness on MTRD data, there is no method designed for partial multi-view data. Moreover, a combination of learning the accurate manifold on each view and learning the accurate consensus manifold should be considered in order to learn the best low-dimensional matrix.

9.4.3 Learning the Consensus Coefficient Matrix Relying on the Consensus Intrinsic Manifold

Data from all views though sampled from different manifolds should lie on the intrinsic manifold, which is the convex hull of different manifolds [1]. The latent low-dimensional data represented by many views should follow the consensus low dimension. To the best of our knowledge, MMNMF [1] is the only method learning the consensus manifold and using it in the NMF framework effectively. The method proposes two objective functions to learn the accurate coefficient matrix relying on the consensus manifold. The consensus manifold is learned beforehand by linearly combining the manifolds learned from all views as

$$L_* = \sum_{v=1}^{n_v} \lambda_v L_v \tag{9.30}$$

The first objective function MMNMF1 is as,

$$J = \sum_{v=1}^{n_v} D_{KL}(X_v || H_v W_v) + \lambda Tr\left(H_*^T L_* H_*\right) \tag{9.31}$$

where $D_{KL}(.)$ is the Kullback–Leibler divergence. The consensus coefficient matrix H_* is constructed by linearly combining all views low-rank matrices as $H_* = \sum_{v=1}^{n_v} \alpha_v H_v$. In this objective function, different coefficient matrices are learned in the new manner to ensure the local geometric structures embedded on each view. Therefore the authors deem that the consensus matrix will be ensured to preserve the diverse geometric structures for the newly mapped space of all views and will return a diverse consensus matrix. Yet, this point is also a disadvantage of the objective function, since it will balance the importance of different views and fails to learn the accurate consensus matrix in the case where some views in the dataset are more important than others.

In the second objective function (MMNMF2), the common coefficient matrix is learned at the same time as different low-rank matrices from all views are learned. The objective function [1] is expressed as

$$J = \sum_{v=1}^{n_v} D_{KL}(X_v || H_v W_v) + \alpha_v D(H_* - H_v) + \lambda Tr\left(H_*^T L_* H_*\right) \tag{9.32}$$

It can be noted that different factor matrices during the learning process are constrained to be as close to the consensus matrix as possible. At the same time, the data points belonging to the consensus low-dimensional space should lie on the convex hull of the multi-manifold. This consensus matrix from this objective

Table 9.1 Characteristics of methods

Methods	Learn the consensus matrix	Learn the manifold on each view	Learn the accurate manifold	Learn the consensus manifold
MVNMF	No	No	No	No
MultiNMF	Yes	No	No	No
STNMF	Yes	Yes	No	No
DRCC	Yes	Yes	No	No
RHCHME	Yes	Yes	Yes	No
ARASP	Yes	Yes	Yes	No
MMNMF	Yes	Yes	No	Yes

function can better reflect the latent clusters shared from all views' data. MMNMF2 can be considered as the well-defined framework for combining NMF and manifold learning since it is able to return the natural consensus shared from all views that respect consensus manifold embedded. MMNMF is believed to be the most effective method since it considers all cases of applying manifold on multi-view data. However in this method, since the consensus manifold is linearly combining all manifold from all views, it may fail to reflect the natural consensus manifold. Similar to learning the consensus coefficient matrix, the consensus manifold should also be learned naturally during the learning process.

The MMNMF method is designed for multi-view data. There are no methods exploiting the consensus manifold designed for MTRD or partial multi-view data. Furthermore, the NMF-based multi-view method utilizing consensus manifold should also be combined with the techniques to learn the accurate manifold on each view in order to achieve the best solution.

We summarize the main characteristics of the three family of methods discussed in this section in Table 9.1.

9.5 Experiments

This section presents empirical analyses of leading NMF-based clustering methods. The clustering performance is evaluated by using criteria of normalized mutual information (NMI) [37], clustering accuracy (AC) and runtime performance. NMI focuses on measuring the similarity between the cluster labels and true classes. The accuracy is the percentage of correctly obtained labels.

Table 9.2 Characteristic of the datasets

Properties	D1	D2	D3	D4	D5
≠ classes	25	10	5	7	17
≠ object types/views	3/2	3/2	3/2	3/2	3/2
≠ samples	1413	1500	187	12,708	617
≠ terms	2921	3000	1702	1433	–
≠ concepts	2437	3000	–	–	–
≠ links	–	–	578	5429	–
≠ actors	–	–	–	–	1398
≠ keywords	–	–	–	–	1878

9.5.1 Datasets

Several well-known multi-view datasets have been used to test the performances of state-of-the-art multi-view clustering methods as shown in Table 9.2.

Reuters-21578[1] is a collection of Reuters newswire stories. The dataset is used popularly in clustering evaluation. We made use of two subsets, R-MinMax (D1) and R-Top (D2) extracted from the dataset. R-MinMax includes 25 classes with at least 20 and at most 200 documents that were extracted from each class and R-Top contains 10 largest classes. For multi-view setting, another view data was created by using extra knowledge, i.e., Wikipedia by following steps in [11, 38] to create a new data view between documents and concepts (along with the first view representations between documents and terms). For MTRD setting, three object types: documents, terms, concepts as well as three inter-type relationships between documents–terms, documents–concepts and term–concepts are used.

Texas[2] (D3) dataset describes web-pages of Texas University. The dataset represented two views: terms and links. For MTRD setting, there are three data types: web-pages, words and links. The clustering task is to achieve different groups of web-pages.

Cora[3] (D4) is a scientific publication dataset with a set of publications represented by two views, contents and sites.

Movie (D5) dataset contains a set of movies represented by two views, actors and keywords. The dataset is used popularly in clustering evaluations such as [31, 39, 40]. For MTRD setting, three object types, movies, actors and keywords are considered together with the relationships that exist between these type objects.

[1] http://www.daviddlewis.com/resources/testcollections/reuters21578/.

[2] http://www.cs.umd.edu/projects/linqs/projects/lbc/.

[3] http://www.cs.umd.edu/%7Esen/lbc-proj/data/cora.tgz.

9.5.2 Benchmarking Methods

The state-of-the-art methods belonging to three family of algorithms, as discussed in Sect. 9.4, are used in evaluation. In the first family of algorithms, DRCC [33] and STNMF [29] have been used. In the second family of algorithms, RHCHME [11] and ARASP [4] have been used. In the third family of algorithms, the only proposed method MMNMF [1] has been used.

DRCC is the co-clustering method, manifold learning is applied on both samples and features. To compare with other MTRD and multi-view methods, we extend DRCC to run on the MTRD dataset. Specifically, we apply manifold learning on every object type to learn the low-rank representations of different object types such that these low-rank factor matrices of every object types will be smooth with its corresponding original manifold.

STNMF is the prominent method designed for MTRD data. In this method, the manifold learning is applied on all data types and is embedded in the symmetric matrix form.

MMNMF [1] learns the consensus coefficient matrix by linearly combining low-rank coefficient matrices of all views, as expressed in Eq. (9.31). For easy comparison with other NMF-based methods, we use Euclidean distance instead of Kullback–Leibler in the cost function Eq. (9.31).

RHCHME [11] learns its complete and accurate manifold by investigating data sampling on manifold and subspace for all object types. The method uses the symmetric matrix form as STNMF [29] to formulate the Laplacian matrices as well as inter-type relationships and factor matrices as in Eqs. (9.20), (9.7), (9.9).

ARASP [4] attempts to learn the accurate manifold by utilizing both close and far distances when constructing the affinity matrices on each object type.

We also include MVNMF and MultiNMF [21] in empirical analyses that are NMF-based clustering methods without manifold learning. These two methods will assist us to check the role of learning the manifold when using NMF in clustering. MVNMF is a special case of MMNMF [1] that ignores considering data lying on manifold by setting the parameter λ to 0 in Eq. (9.31). The consensus matrix is calculated as a linear combination of all coefficient matrices. MultiNMF [21] learns the low-rank representations of data on different views at the same time the consensus coefficient matrix is learned, without manifold learning.

The optimizing technique used in all benchmark methods in this paper is the multiplicative update rule (MUR). The technique has been established in [34] where the optimizing process will separately and sequentially update each variable in the objective function while keeping other variables as constants until convergence.

9.5.3 Clustering Performances

Tables 9.3 and 9.4 list the clustering results between the three families of algorithms. The leading first family of algorithms, DRCC and STNMF learn the manifold of data on every view and maintain the manifolds when learning data from high to low dimension. Effect of this learning can be seen by their superior performance over MultiNMF and MVNMF that ignore learning the manifold of data.

The methods from the second family, ARASP and RHCHME, not only learn the manifold on every view but also learn the low-dimensional embeddings based on learning the accurate manifold on each view. Since focusing on learning the accurate manifold, this group of methods outperforms all other methods on all benchmark datasets. For RHCHME, the superior performance comes from learning the diverse manifolds by considering the data lying on manifolds as well as data belonging to subspaces. The manifold is accurate in the sense that it is built based on the affinity matrix that includes all useful closeness information. ARASP learns the accurate manifold by incorporating both close and far distances information between data points, thus it can maintain not only the local geometric structure but also the global geometric structure of data.

The third family of algorithm that learns the consensus manifold is believed to bring the good performance since the assumption about the consensus manifold is promising. However, as seen from the clustering results, MMNMF achieves poorest results majority of times. This may be due to the fact that the cluster structures of different views can be similar; however, the geometric structure of original views

Table 9.3 NMI for each methods and datasets

Methods	D1	D2	D3	D4	D5	Average
MVNMF	62.59	57.19	21.02	29.11	15.60	**37.10**
MultiNMF	69.40	55.09	20.65	30.49	20.87	**39.30**
STNMF	67.28	57.89	31.35	20.20	31.34	**41.61**
DRCC	72.71	59.68	35.78	31.66	33.93	**46.75**
RHCHME	70.36	62.04	33.15	33.53	31.34	**46.08**
ARASP	75.01	69.33	39.23	35.24	33.18	**50.40**
MMNMF	66.80	58.54	20.23	18.32	22.99	**37.38**

Table 9.4 Accuracy for each methods and datasets

Methods	D1	D2	D3	D4	D5	Average
MVNMF	53.14	61.33	51.87	53.03	17.18	**47.38**
MultiNMF	52.80	64.40	53.48	46.16	23.01	**47.97**
STNMF	51.38	63.87	44.92	41.29	26.58	**45.61**
DRCC	57.32	59.13	62.57	50.33	32.58	**52.39**
RHCHME	65.18	64.15	50.27	55.28	31.44	**53.26**
ARASP	71.05	73.33	59.36	58.31	33.39	**59.09**
MMNMF	52.51	61.13	38.50	33.97	20.91	**41.40**

may be different. Constraining the low-rank representations of different views to be smooth with the common consensus manifold may cause the bad effect to the latent features learning. We suppose that (1) the assumption of the consensus manifold can only be used in the ideal situation when the data represented from different views is sampled on similar structures or manifolds or (2) the intrinsic accurate consensus manifold should be learned in a different way rather than linearly combining as in MMNMF [1].

The two NFM-based clustering methods without manifold learning, MVNMF and MultiNMF, achieve poor results since they learn the consensus matrix without checking on how the original data is distributed in space. On most datasets, MultiNMF performs better than MVNMF since the former method learns the consensus matrix simultaneously with learning the low-rank matrices of all views. Whereas MVNMF learns the consensus matrix after all factor matrices of different views are learned, hence, the consensus matrix is simply a linear combination of all views low-rank matrices.

With regard to time complexity (Table 9.5), we observed that STNMF, RHCHME and MMNMF consume more running time than the other methods. STNMF and RHCHME require some time to process on big symmetric matrix form and MMNMF uses much time to ensure each low-rank representation of each view respecting the consensus manifold of the data. ARASP consumes the least time and enjoys high accuracy since it makes use of the pseudo orthogonal constraint (i.e., the normalized-cut type constraint) for the fast convergence [41].

9.5.4 Parameters Setting

To equally compare all methods on all datasets, we set the same ranges of sharing parameters for all datasets and report the best results for each method on each dataset.

Specifically, except for MultiNMF and MVNMF, there are two common parameters for all methods, i.e., the manifold regularization parameter λ and the neighbourhood size k. For λ, we search in the range $\lambda = \{0, 1, 0.5, 1, 10, 50, 100\}$. For k, we search in the range $k = \{5, 10, 20, 30, 50, 70, 100\}$. The values of these

Table 9.5 Running time for each methods and datasets

Methods	D1	D2	D3	D4	D5	Average
MVNMF	77	215.48	7.8	319	18	**127**
MultiNMF	120.5	277	14	157	31	**120**
STNMF	220	240	16	136	57	**134**
DRCC	200	190	14	164	54	**124**
RHCHME	220	250	16	139	58	**136**
ARASP	140	100	12	74	40	**73**
MMNMF	262.68	209	5.7	274	63	**163**

Table 9.6 Parameter for each method on each dataset

Methods	Parameter	D1	D2	D3	D4	D5
STNMF	λ	10	10	100	20	1
	k	40	20	30	5	20
DRCC	λ	1	10	50	20	1
	k	30	30	35	15	70
RHCHME	λ	10	20	10	20	20
	k	40	10	10	5	30
ARASP	λ	20	10	1	20	1
	β	10	20	1	10	1
	k, p	40	45	45	70	35
MMNMF	λ	0.1	0.1	0.1	1	0.1
	k	5	20	20	5	5

parameters that allow each method to achieve the best result on each dataset are given in Table 9.6. For example, RHCHME and STNMF achieve the best results when $\lambda = 10$ on dataset D1 and when $\lambda = 20$ on dataset D4.

Apart from λ and k, RHCHME uses another parameter, i.e., α in Eq. (9.25) to control the contribution of considering data lying on manifold or subspace. For simplicity, we set $\alpha = 1$ to let the two kinds of manifolds play the same role in the manifold learning process.

In ARASP, there are two more parameters, β and p to set the role of far distance information when constructing the affinity matrix on each data type in Eq. (9.26). To set β, we fix the value for λ and let β value selecting such that $\beta/\lambda = [0.1 \ldots 1]$. In our experiment, ARASP reaches its peaks on all datasets on any value in the range. We choose the farthest neighbourhood size p runs in the same manner as range with the nearest neighbourhood size.

For MMNMF, the parameter λ_i to construct the consensus manifold in Eq. (9.30) is set to be $1/n_v$, i.e., every manifold of every view play the same role in the consensus manifold.

9.6 Discussion and Potential Work

As evidenced by the experiment results, manifold learning plays a vital role in clustering. The reason lies on the ability to learn the intrinsic structure of data which helps to discover the latent cluster structures. However, how to effectively apply manifold learning on multi-view data is still an unsolved problem. Next we discuss some of these limitations and future works.

Choosing the Appropriate Neighbourhood Size *k*

The problem of choosing the appropriate neighbourhood size k in kNN graph-based method is still haunting [7, 12, 42]. Manifold learning is based on constructing the

kNN graph, thus it also suffers from the problem of choosing k. ARASP constructs the affinity matrix in a novel fashion by combining both close and far distances, thus reduces the dependence of constructing the affinity matrix in the neighbourhood size k. Similarly, RHCHME reduces the dependence on k by considering both data lying on manifold and subspace, thus it ensures the affinity matrix to include all useful neighbours. Though ARASP and RHCHME have provided effective techniques to learn the affinity matrix with less dependence on k, it still needs more attention.

Another issue associated with using kNN graph is its poor performance with high-dimensional datasets. In high-dimensional data space, the concept of distance diminishes due to high sparsity and the k nearest neighbours start to approach the far points [43]. Consequently, the kNN graph will become unstable or the affinity matrix will be meaningless.

The Consensus Manifold
The idea that the consensus manifold should exist in multi-view data is novel and promising. However, the consensus manifold should be learned in a different way rather than linearly combining as in [1]. This area needs much needed attention.

MTRD Data Verse Multi-View Data
We observed that the MTRD data provides higher performance as compared to multi-view data. As can be seen in experiment result, methods applied on MTRD data such as DRCC, STNMF, RHCHME and ARASP achieve more accurate clusters than methods designed for multi-view data. This is explained by the fact that MTRD utilized more information than multi-view data. For instance, while multi-view method applied on D1 can only use two views of data represented by two data matrices such as document-term and document-concept, MTRD clustering methods can make use of the relationships between terms and concepts. This proves the fact that the more information used, the higher results yielded.

Integrating Data from Multi-View
As discussed in the paper, there are two forms of fusing data in multi-view problem, one is using NMF and the other is using coupled matrix factorization. Since the coupled matrix factorization aims to learn the consensus coefficient matrix during the learning process instead of learning separate coefficient matrices of all views, the assumption about the manifold on each view and the consensus manifold will be different. Investigating how the coupled factorizing process should respect the intrinsic manifold is promising for future work to obtain a better outcome.

9.7 Conclusion

The paper presented a survey on the class of methods using manifold learning on the NMF framework for multi-view data. Different categories of methods have been surveyed as well as many instances of multi-view data such as partial multi-view data or multi-type relational data have also been investigated.

In future works, some characteristics of data such as the density of data points and the number of dimensions should be taken into account to investigate how characteristic of data affects the effectiveness of using a manifold. How to learn the accurate consensus manifold is also a promising issue to be paid attention.

References

1. Zhang, X., Zhao, L., Zong, L., Liu, X., Yu, H.: Multi-view clustering via multi-manifold regularized nonnegative matrix factorization. In: Proceedings of the International Conference on Data Mining, pp. 1103–1108 (2014)
2. Qi, L., Shi, Y., Wang, H., Yang, W., Gao, Y.: Multi-view subspace clustering via a global low-rank affinity matrix. In: Yin, H., Gao, Y., Li, B., Zhang, D., Yang, M., Li, Y., Klawonn, F., Tallón-Ballesteros, A.J. (eds.) Intelligent Data Engineering and Automated Learning – IDEAL 2016, pp. 321–331. Springer International Publishing, Cham (2016)
3. Parsons, L., Haque, E., Liu, H.: Subspace clustering for high dimensional data: a review. SIGKDD Explor. Newsl. **6**(1), 90–105 (2004)
4. Luong, K., Nayak, R.: Learning association relationship and accurate geometric structure for multi-type relational data. In: Proceedings of the International Conference on Data Engineering (2018)
5. Lee, D., Seung, H.: Algorithms for non-negative matrix factorization. In: Advances in Neural Information Processing Systems 13, pp. 556–562. MIT Press, Cambridge (2001)
6. Lee, D., Seung, H.: Learning the parts of objects by non-negative matrix factorization. Nature **401**(6755), 788–91 (1999)
7. Cai, D., He, X., Han, J., Huang, T.S.: Graph regularized nonnegative matrix factorization for data representation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(8), 1548–1560 (2011)
8. Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. J. Mach. Learn. Res. **7**, 2399–2434 (2006)
9. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**(6), 1373–1396 (2003)
10. Tenenbaum, J.B., Silva, V.D., Langford, J.C. A global geometric framework for nonlinear dimensionality reduction. Science **290**(5500), 2319–2323 (2000)
11. Hou, J., Nayak, R.: Robust clustering of multi-type relational data via a heterogeneous manifold ensemble. In: Proceedings of the International Conference on Data Engineering (2015)
12. Li, P., Bu, J., Chen, C., He, Z.: Relational co-clustering via manifold ensemble learning. In: Proceedings of the ACM International Conference on Information and Knowledge Management, pp. 1687–1691 (2012)
13. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2, ICCV '99, p. 1150. IEEE Computer Society, Washington, DC (1999)
14. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1, pp. 886–893 (2005)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, NIPS'12, pp. 1097–1105. Curran Associates, Red Hook (2012)
16. 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)

17. Winn, J., Jojic, N.: Locus: learning object classes with unsupervised segmentation. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, pp. 756–763 (2005)
18. Li, S.-Y., Jiang, Y., Zhou, Z.-H.: Partial multi-view clustering. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, pp. 1968–1974. AAAI Press, Palo Alto (2014)
19. Rai, N., Negi, S., Chaudhury, S., Deshmukh, O.: Partial multi-view clustering using graph regularized NMF. In: 2016 23rd International Conference on Pattern Recognition (ICPR), pp. 2192–2197 (2016)
20. Qian, B., Shen, X., Gu, Y., Tang, Z., Ding, Y.: Double constrained NMF for partial multi-view clustering. In: 2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA), pp. 1–7 (2016)
21. Jing, G., Jiawei, H., Jialu, L., Chi, W.: Multi-view clustering via joint nonnegative matrix factorization. In: SDM, pp. 252–260. SIAM, Philadelphia (2013)
22. Wang, J., Tian, F., Wang, X., Yu, H., Liu, C.H., Yang, L.: Multi-component nonnegative matrix factorization. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17, pp. 2922–2928 (2017)
23. Wang, J., Tian, F., Yu, H., Liu, C.H., Zhan, K., Wang, X.: Diverse non-negative matrix factorization for multiview data representation. IEEE Trans. Cybern. **PP**(99), 1–13 (2017)
24. Zhang, X., Gao, H., Li, G., Zhao, J., Huo, J., Yin, J., Liu, Y., Zheng, L.: Multi-view clustering based on graph-regularized nonnegative matrix factorization for object recognition. Information Sciences **432**, 463–478 (2018)
25. Luong, K., Balasubramaniam, T., Nayak: A novel method of using coupled matrix and greedy coordinate descent for multi-view data representation. In: Proceedings of the 19th International Conference on Web Information Systems Engineering (2018)
26. Acar, E., Gurdeniz, G., Rasmussen, M.A., Rago, D., Dragsted, L.O., Bro, R.: Coupled matrix factorization with sparse factors to identify potential biomarkers in metabolomics. Int. J. Knowl. Disc. Bioinform. (IJKDB) **3**(3), 22–43 (2012)
27. Acar, E., Bro, R., Smilde, A.K.: Data fusion in metabolomics using coupled matrix and tensor factorizations. Proc. IEEE **103**, 1602 (2015)
28. Ding, C., Li, T., Peng, W., Park, H.: Orthogonal nonnegative matrix t-factorizations for clustering. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data mining, pp. 126–135 (2006)
29. Wang, H., Huang, H., Ding, C.: Simultaneous clustering of multi-type relational data via symmetric nonnegative matrix tri-factorization. In: Proceedings of the ACM International Conference on Information and Knowledge Management, pp. 279–284 (2011)
30. Xu, W., Liu, X., Gong, Y.: Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 267–273 (2003)
31. Cai, D., He, X.: Manifold adaptive experimental design for text categorization. IEEE Trans. Knowl. Data Eng. **24**(4), 707–719 (2012)
32. Geng, B., Xu, C., Tao, D., Yang, L., Hua, X.S.: Ensemble manifold regularization. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2396–2402 (2009)
33. Gu, Q., Zhou, J.: Co-clustering on manifolds. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 359–368 (2009)
34. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 269–274. ACM, New York (2001)
35. George, T., Merugu, S.: A scalable collaborative filtering framework based on co-clustering. In: Proceedings of the Fifth IEEE International Conference on Data Mining, ICDM '05, pp. 625–628 (2005)
36. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
37. Zhong, S., Ghosh, J.: Generative model-based document clustering: a comparative study. Knowl. Inf. Syst. **8**(3), 374–384 (2005)

38. Jing, L., Yun, J., Yu, J., Huang, J.Z.: High-order co-clustering text data on semantics-based representation model. In: Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 171–182 (2011)
39. Bisson, G., Grimal, C.: Co-clustering of multi-view datasets: a parallelizable approach. In: 2012 IEEE 12th International Conference on Data Mining, pp. 828–833. IEEE, Piscataway (2012)
40. Hussain, S.F., Mushtaq, M., Halim, Z.: Multi-view document clustering via ensemble method. J. Intell. Inf. Syst. **43**(1), 81–99 (2014)
41. Gu, Q., Ding, C., Han, J.: On trivial solution and scale transfer problems in graph regularized NMF. In: Proceedings of the International Joint Conference on Artificial Intelligence, pp. 1288–1293 (2011)
42. Li, Z., Liu, J., Lu, H.: Structure preserving non-negative matrix factorization for dimensionality reduction. Comput. Vis. Image Underst. **117**(9), 1175–1189 (2013)
43. Beyer, K.S., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: Proceedings of the 7th International Conference on Database Theory, ICDT '99, pp. 217–235. Springer, London (1999)

Chapter 10
Leveraging Heterogeneous Data for Fake News Detection

K. Anoop, Manjary P. Gangan, Deepak P, and V. L. Lajish

Abstract Nowadays, a plenty of social media platforms are available to exchange information rapidly. Such a rapid propagation and cumulation of information form a deluge, in which it is hard to believe all the pieces of information since it appears to be very realistic. In this context, characterizing and recognizing misinformation, especially, fake news, is a highly recommended computational task. News fabrication mostly happens through the textual and visual content comprised in the news article. People spreading fake news have been intentionally modifying the content of a news with some partially true information or use fully manipulated information, newly fabricated stories, etc., which could mislead others. Fake news characterization and detection are the computational studies that focus to get rid of the highly malicious news creation and propagation. The textual and visual content-related features, temporal and propagation patterns of the network, that use traditional and deep neural computations are the methods to identify fake news generation and spread. This chapter discusses the methods to leverage heterogeneous data to curb the fake news generation and propagation. We present an extensive review of the state-of-the-art fake news detection systems, in the context of different modalities emphasizing the content-based approaches including text and image modality and also discuss briefly the network, temporal, and knowledge base approaches. This study also extends to discuss the available datasets in this area, the open issues, and future directions of research.

K. Anoop · M. P. Gangan · V. L. Lajish (✉)
Department of Computer Science, University of Calicut, Malappuram, Kerala, India
e-mail: anoopk_dcs@uoc.ac.in; manjaryp_dcs@uoc.ac.in; lajish@uoc.ac.in

Deepak P
Queen's University Belfast, Belfast, UK
e-mail: deepaksp@acm.org

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_10

10.1 Introduction

The rapid growth in technology, the advent of internet, and digitization of media has minimized the challenges with respect to the geographic reach to the source of news and quick delivery of information. Besides the traditional news media, online social network serves as the major medium to share news information along with user opinion. There are many studies reporting the role of social networks in real-world events, like assistance, situational awareness and support during disasters, crisis, and emergencies [1–7]. But, the success of technological advancements and digitization in news media is harmed by the creation and propagation of fake information.

Fake news is a category of mis- or disinformation that spreads through the traditional news media or online social media to mislead people. Misinformation deals with the unintentional spread of false information, whereas disinformation is intentional [8, 9]. There are various categories of mis- or disinformation [10, 11]. *False connection* is one category where the news headline, caption, or the images in the news do not correspond to the news content. Another category called *false context* shares genuine content in false context. *Manipulated contents* are produced by falsifying the genuine content or image in the news. *Misleading contents* misuse some information to frame a different issue with the false claim. *Imposter contents* impersonate the genuine sources. A hundred percentage false news content to deceive people comes under the category of *fabricated content*. *Satire*, *parody*, or *trolls* comes in another category that is to just amuse or fool readers and has no harmful intentions.

The creation of fake information is not new and dates back to early thirteenth century BC where Ramasses II spread lies and propaganda on victory of Egyptian Empire over Hittite Empire in the battle of Kadesh [12]. Nowadays also, news is contaminated by various sorts of fake content which makes people bewildered in the genuineness of news reaching them. Since fake news was one of the greatest issues of 2016, the Macquarie Dictionary Word of the Year 2016[1] is *fake news*. Similarly, *post-truth* was the Oxford Dictionaries Word of the Year 2016.[2] A recent interesting research finding at MIT reports that fake news diffuses significantly faster, farther, deeper, and broader than a real news, as a result of user retweets on the Twitter network [13]. They find that fake news appears more novel than the real ones [14].

The inducement of fake information in news can be for several reasons like partisanship, provoking on a topic, political influence or power, a means to earn the profit, harm an agency or person, etc. The fake information inducers use sensational and fabricated captions for increasing the reads, share, or internet click revenue. An example is Click-bait, a weblink to generate advertising revenue, by providing such sensational headlines to make the readers curious to click the linked content.

[1] https://www.macquariedictionary.com.au/news/view/article/431/.

[2] https://en.oxforddictionaries.com/word-of-the-year/word-of-the-year-2016.

10.1.1 Impact of Fake News on Society

There have been many instances highlighting the inimical effects of fake news on real-life events. The fake news regarding the natural disasters proliferating via online social media has lead panic and chaos among people [15]. There are studies on the role of social media to spread fake news during natural disasters, like in the case of Hurricane Sandy during 2012 [16] (Fig. 10.1), the earthquake in Chile during 2010 [17], and Japanese Earthquake and Tsunami during 2011 [18]. False information also has adverse effects on the large-scale investments and stock prices. One such example is the fake tweet saying about an explosion that injured Barack Obama that wiped out $130 billion in stock value within a few minutes following a tweet [19]. Rumors are also prominent in the area of health and disease [20, 21], terrorists attacks [22], etc. Apart from these areas, fake content is more profound in the political news to manipulate the public events like elections [23–28]. These ramifications of rapidly propagating fake news through social spaces on individuals and society include the injection of biased or false beliefs to the consumers, without their awareness, that results in changing their interpretations and response to a real news. In effect, this disturbs the authenticity of the whole news ecosystem.

In this chapter, we provide an extensive survey of fake news detection methods leveraging heterogeneous data. Section 10.2 describes fake news detection methods and modalities with a special emphasis on content-based methods utilizing text and image content. This section also briefly discusses the network and temporal modalities for fake news detection. Section 10.3 lists out some of the available datasets for fake news detection-related research, and Sect. 10.4 discusses the open issues and future directions of research in fake news detection.

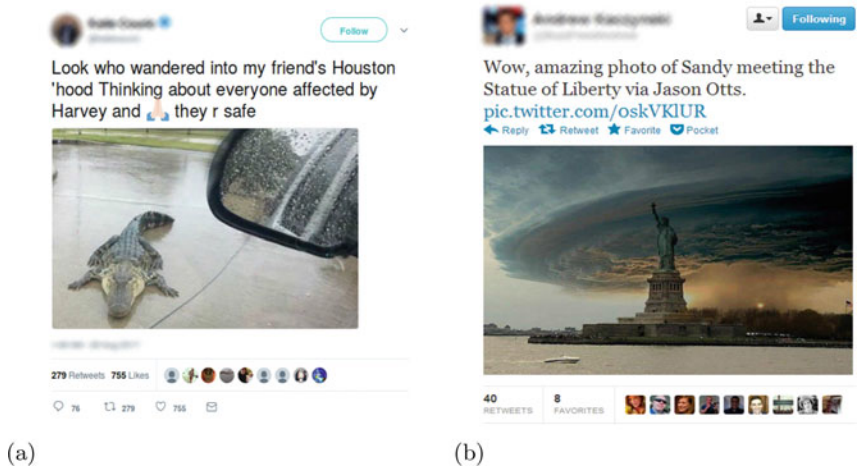


Fig. 10.1 Fake information tweets on Hurricane Sandy. (**a**) An alligator in Houston during Hurricane Harvey. (**b**) New York harbor with Frankenstorm bearing down

10.2 Fake News Detection

Fake news detection systems employ techniques to identify the genuineness of news through the perspectives of the text and image content in the news, social network, temporal features, and knowledge base or human-in-the-loop methods. In general, the content-based approaches consider the text and image modalities modeled using several machine learning techniques namely unsupervised, semi-supervised, and supervised learning, including deep learning techniques. In *textual modality*, the features of interest are stylometric, linguistic, structure and syntax based, statistical, etc. In case of *image modality*, the focus is on text associated with the image, temporal information of the posted image, or image forensic features to detect tampering. User-based features, propagation features, structure and behavior of the network, etc. are those considered in the domain of *social network-based* fake news detection studies. To discern fake news, works are also reported using *temporal information* of users, events, or articles. Fake news detection methods also contain *knowledge base approaches* including human-in-the-loop methods to discuss the worth of crowdsourcing techniques and fact-checkers that distinguish the fake and real news. The overall architecture of fake news detection in the perspective of modality and learning techniques utilized is shown in Fig. 10.2.

10.2.1 Textual Modality

The prevailing way of characterizing and detecting fake news relies on the content of the news articles. In general, the content of a news article can be categorized into

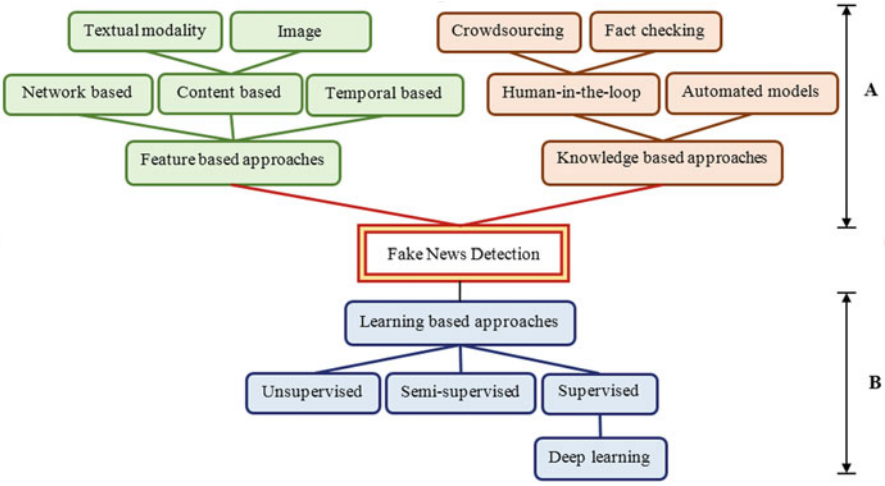


Fig. 10.2 Fake news detection approaches (A: Modalities) (B: Learning techniques)

textual and visual modality. Among these modalities, a major portion of information in the news is populated with the help of textual modality. Following are the sample list of text captions from fake news articles reported in two different fact-checking portals (Snopes and PolitiFact). Where the headlines in list 1–3 and its corresponding article is fully fake and 4th one is half true.

1. Fluoride Officially Classified as a Neurotoxin in World's Most Prestigious Medical Journal [29]
2. SHOCKER!!!! A University female student confesses to have infected more than 324 men in Nakuru with HIV. She targets 2000 before the year ends!!!! [30]
3. President Trump Announces Plans To Build A Monument Commemorating the War on Christmas [31]
4. The United States has a massive trade deficit with Japan. It's anywhere from $69 billion to a $100 billion a year [32]

A bunch of research approaches to detecting fake news relies on characterizing and analyzing the patterns in textual data of the fake news [33–36]. Considering the vast availability and usefulness of textual data, for detecting fake news, researchers are more interested to provide their own news corpus which includes both the legitimate and fake news article using crowdsourcing and fact-checking methods. Andreas Vlachos et al., in his research study, proposes the general way of constructing a benchmark dataset for fake news detection or checking the fact of a social media news [37]. In order to create the textual corpus, they approached two major fact-checkers websites including the Channel4[3] and PolitiFact.[4] According to Andreas Vlachos et al., this fact-checking is a traditional supervised classification task. They find that rather than traditional binary classification, fake news detection is multiclass, and they come up with a fine-grained labeling scheme on a five-point scale, which includes TRUE, MOSTLY TRUE, HALF TRUE, MOSTLY FALSE, and FALSE label for each news articles. Also, the study mentioned another approach of semantic similarity between statements to explore fake news detection and automated fact-checking.

William Yang in his research comes up with the creation of another benchmark dataset for fake news detection and also tries to experiment the possibilities and effectiveness of both traditional and the deep models of machine learning algorithms in fake news detection [38]. LIAR is the publicly available benchmark dataset, which includes 12.8 k human-labeled short statements from PolitiFact, and the articles are mapped into six fine-grained labels, including PANTS-FIRE, FALSE, BARELY TRUE, HALF-TRUE, MOSTLY TRUE, and TRUE. Because of the six fine-grained labels, the author views the problem as a simple 6-way multi-class classification problem. In this approach, the experiment of fake news detection is carried out using five baselines including a majority baseline, traditional Logistic Regression (LR), Support Vector Machine (SVM), Bi-directional Long Short-Term

[3] http://blogs.channel4.com/factcheck/.

[4] http://www.politifact.com/.

Memory network model(Bi-LSTM), Convolutional Neural Network (CNN), and a Hybrid CNNs. Except for the hybrid CNNs, all other models are built only with textual data, and the hybrid CNNs is built with the combination of text and other metadata details like subject, speaker, job, state, contexts, party, and history. The results are measured as validation and test set accuracy. The text classifiers such as SVM and LR models obtained significant results. Due to overfitting, Bi-LSTM did not perform well. CNN outperformed all models on the held-out set.

10.2.1.1 Textual Modality Features for Fake News Detection

Linguistic Features

The traditional representations of linguistic features used for modeling documents are Lemmas (base form of morphological categories like nouns, e.g., Hospital from Hospitals, or verbs, e.g., dance from danced, or dancing), phrases (the sentence pieces as word sequences), and word senses (different meanings of content words, as defined in dictionaries) [39].

N-grams

An N-gram model is defined as a probabilistic language model widely used in computational linguistics (e.g., statistical natural language processing) for predicting the next item in a contiguous sequence of N items from a given sample of text or speech. In general, N-gram is defined as a set of co-occurring words within a given window. For example, for a given sentence "Did a Trump Tower Open in Pyongyang," if $N = 2$, then "Did a," "a Trump," "Trump Tower," "Tower Open," "Open in," "In Pyongyang," are the N-grams. An N-gram of size 1 is referred to as a *unigram*; size 2 is referred to as a *bigram* (or, less commonly, a *digram*); size 3 is referred to as a *trigram*, and so on. To model several features from this N-gram representation, the easiest way is to use the bag of word representations and word length features encoded as tf-idf values.

POS

The eight parts-of-speech tagging including noun, verb, pronoun, preposition, adverb, conjunction, participle, and article are another way of modeling the hidden language patterns in the text documents and which can be highly utilized for the study of fake news detection.

Punctuation

Punctuation characters such as period, comma, dashes, question marks, and exclamation marks are also linguistic features that might be effective in categorizing the deceptive text from the truthful text. Several researchers report that this punctuation

shows considerable influence on fake news detection and even in opinion spam detection [40, 41]. Rubin et al. propose that a greater number of clauses increase the number of punctuation marks found in satire, which informed the development of punctuation features such as period, commas, dashes, question marks, and exclamation marks with the help of the Linguistic Inquiry and Word Count Software [40]. According to Veronica et al., the authors of fake news use more adverbs, verbs, and punctuation characters than the authors of legitimate news [33].

Syntactic and Semantic Features

The syntactic representation of sentences in language also influences the detection of fake and real news articles. The conventional way of observing the syntactic representation among the language is by using several statistical parsers. Badaskar et al. in his previous research work hypothesized that real sentence tends to be grammatical, while the same may not be the case for fake sentences [42]. This hypothesis directs the proposed work into a way of measuring the grammaticality using a statistical parser. The log-likelihood score returned by the statistical parser can be used to judge the grammaticality of a sentence and thus determine whether it is fake or real. In another research work, the author captures the syntax feature derived from production rules based on a context-free grammar tree. The parser used for the extraction of these syntactic features is Stanford Parser [43].

The semantics of an article can be observed through the inter- and intrarelationship between correlated pairs of content words and sentences. The real articles comprise the correlated pairs of content words and sentences that correlate with each other, and this correlation is said to be semantically coherent [42]. There are also two different types of coherence reported in the literature, which are intra-sentence coherence for modeling the intra-sentence word correlations and inter-sentence coherence to model the topical redundancy.

Readability

Text understandability or readability is yet another feature for fake or deception understanding. In general, this includes features like number of characters, complex words, long words, number of syllables, word types, number of paragraphs, and several other readability metrics, including the Flesh-Kincaid, Flesch Reading Ease, Gunning Fog, and Automatic Readability Index [33].

Psycholinguistic Features

Psycholinguistics or psychology of language is used to extract the psychological characteristics of language, including the emotions embedded, self-references, cognitive complexity, etc., from a bit of text data. Matthew L. et al. in his research work reported that the deceptive language shows characteristics of fewer self-references, more negative emotion words, and fewer markers of cognitive complexity [35]. In

contradiction to this work of Matthew L. et al., Catalina L. et al. propose a study on deception detection on dating profiles using linguistic cues [44], which reports the presence of fewer negative emotions in the deceptive dating profile. The study of Catalina L. et al. also illustrates that profile deceptions correlate with fewer self-references, increased negations, and fewer overall words, and throughout the study, several hypotheses are formed mainly using the emotional and cognitive linguistic cues.

Other relevant psycholinguistics feature sets used in different studies are the presence of affective information (positive and negative emotion), exclusive information (but, without), motion words (walk, move, go), social processes (talk, us, friends), cognitive processes (cause, know), etc.

Linguistic Inquiry and Word Count (LIWC) is the most popular software used for experimenting with the psycholinguistics characteristics of a text corpus [45]. LIWC is based on a large lexicon of word categories that represent both linguistic and psycholinguistic process including the emotional, social as well as the part-of-speech tagging and several other kinds of word counts. Most of the research work in both computational perspective and social science perspective use this LIWC to evaluate their hypothesis [35, 41, 44, 46].

Stylometric Features

Stylometry is the statistical analysis of variations in literary style between one writer or genre and another. Several stylometric features are also popularly used in the detection of deception in the textual corpus. In general, the field of stylometry uses combinations of linguistic features and machine learning to model the writing style in an article. Afroz et al. in his research argue that some linguistic features change when people hide their writing style and if we identify that features, it is a way of detecting deception through stylistic features [47]. The major contribution of this work is a method for detecting stylistic deception in written documents.

The writeprints feature set is a set of feature that can represent the author's writing style, which is proposed by Zheng et al. in his research study [48]. This writeprints feature set includes several categories like character related (total characters, the percentage of digits, percentage of letters, the percentage of uppercase letters, frequency of character unigram, most common bigrams and trigrams, etc.), frequency of special characters (∼, \, ⟨, ⟩,], [, }, {, $, #, &, etc.) and punctuations, word related (total words, number of characters per word, frequency of large words, most frequent word uni-/bi-/ tri-grams, etc.), and function words and parts-of-speech (frequency of function words and parts-of-speech). More specifically, these feature sets can be grouped into lexical, syntactic, and content-specific features.

In addition to the writeprints feature sets, Afroz et al. experiment the effectiveness of two other set of feature namely lying-detection feature set and authorship-attribution features in [47]. The lying-detection feature set includes quantity (number of syllables,number of words, and number of sentences),

vocabulary complexity (number of big words, and number of syllables per word), grammatical complexity (number of short sentences, number of long sentences, Flesh–Kincaid grade level, average number of words per sentence, sentence complexity, and number of conjunctions), uncertainty (number of words express certainty, number of tentative words, and modal verbs), specificity and expressiveness (rate of adjectives and adverbs, and number of effective terms), and verbal non-immediacy (self-references, and number of first-, second-, and third-person pronoun usage). The authorship-attribution features are the number of unique words, complexity, Gunning-Fog readability index, character count without whitespace, character count with whitespace, average syllables per word, sentence count, average sentence length, and Flesch–Kincaid readability score. The authors continue the study as a supervised classification task using Support Vector Machine (SVM) with Sequential Minimal Optimization (SMO).

Feng et al. propose an unconventional approach of syntactic stylometry for deception detection using features driven from Context Free Grammar (CFG) parse trees, and the results show consistent improvement in performance over several baselines approaches [36]. The experiments are conducted using four different datasets scraped from product review and essay domain.

Statistical or Empirical Features

The statistical or empirical feature is also very helpful to characterize and understand the hidden patterns in fake or real articles when we consider the fake or real article detection as traditional supervised or unsupervised learning processes. In a broader perspective, these statistical or empirical features can be classified as traditional Bag-of-Words (BoW) approach including the Term Frequency and Inverse Document Frequency (TF-IDF) and the latest nontraditional neural word embedding-based features (Word2Vec, Doc2Vec, and Glove).

Bag-of-Words (BoW)

TF-IDF is a numerical statistics which focusses on how a word is important to the document in a corpus. In general, TF-IDF is the product of Term Frequency (TF) weights and Inverse Document Frequency (IDF) weights.

Term frequency ($TF_{t,d}$): number of times the term t occurs in document d.

Inverse document frequency ($IDF_{t,D}$): it is the measure of how much information that a word gives in the document collection, and measures whether the word is rare or common across the document corpus.

Badaskar et al. in a research study mention that the count of uncommon pairs of words within an article, the ratio of the perplexity of trigram and quad-gram models for a given article, and the nature of the POS tags that occur at the start and end of sentences in an article also belong to the category of statistical features, which can be generally computed with the help of BoW and co-occurrence matrices [42].

Neural Word Embedding

The traditional distributional features have been replaced from many NLP architectures by the advent of neural word embedding approach. Unsupervised learning of this word embedding illustrates the semantic relation between several words in the text corpus, for example, $king - man + woman \approx queen$. Mikolov et al. introduce the three-layer neural architecture for the neural word embedding, including two different models, the continuous bag of words (CBOW) and skip gram model [49, 50]. Initially, we feed the one-hot vectors from the text corpus to the embedding input layer and then the neurons in this layer are mapped into a fully connected intermediate layer, then finally a softmax layer that produces a probability distribution over words in the vocabulary. Similar architectures are proposed by several researchers to improve the performance and efficiency; they are Doc2Vec [51] and Global Vectors for Word Representation (GloVe) [52] vector representations.

10.2.1.2 State-of-the-Art

A huge amount of research is carried out in this area of textual modality-based deceptive or fake news recognition using the abovementioned different feature sets. Perez-Rosas et al. in his research focus on two perspectives of automatic fake news detection. First one is the creation of a novel dataset covering different news domains and then learn the fake news patterns using a linear SVM classifier with fivefold cross-validation [33]. To build the supervised learning model, Perez-Rosas et al. only use the linguistic features including N-grams, punctuations, psycholinguistic features, readability, and the syntax features. The author also presents a corresponding analysis of the automatic and manual detection of fake news.

A stylometric-based study for detecting hyperpartisan and fake news is proposed by Potthast et al. This work preliminarily focuses on a large corpus creation which includes 1627 articles that were manually fact-checked from BuzzFeed [34]. In this study, Potthast et al. try to reveal that hyperpartisan left- and right-wing news have a lot more in common than any of the two have with the mainstream. Also concludes the research findings using the stylometric approach that hyperpartisan news can be discriminated well with $F1$ score 0.78 and satire from with an $F1$ score 0.81. Contradictory to the previous $F1$ scores, the experimental results show that style-based fake news detection does not live up to scratch with an $F1$ score of 0.46.

Rubin et al. propose another dimension that satirical cues are used for recognizing possibly misleading news articles [40]. Generally, the author illustrates the basic concepts of satire and humor in the first part of this paper and then explains the features of satirical news which imitate the style and format of journalistic reporting. An SVM model built on 360 news articles using five predictive features, like absurdity, humor, grammar, negative emotions, and punctuations, is the major architecture proposed in this study. Among the five different features, experimental

results report that absurdity, grammar, and punctuations are the best predicting features for recognizing satirical news with an F-score of 87%.

On the other side of traditional learning approach, Bhatt et al. propose deep learning architecture for fake news detection by combining neural, statistical (TF), and other external features [53]. In this study, the unrelated headline–body pair FNC-1 dataset is used for exploring a subtask of fake news detection and that is stance detection. The major focus is on classifying the headline–body pairs into agree, disagree, discuss, or unrelated classes. This study demonstrates a skip-though vector for a neural feature, which will encode the sentence into a 4800-length vector embedding. The number of characters n-gram match between headline–body pair and the number of word n-gram match between headline–body pair, weighted TF-IDF score between headline–body pair, etc. are the handcrafted external feature combination included in this study. At last, the proposed method uses a deep neural layer for classification of the combined features.

Chopra et al. propose a different approach to fake news recognition via stance detection with deep learning models [54]. Similar to the previous research findings of Bhatt et al., here also the authors use Fake News Challenge I (FCN-I) Headline–Article dataset towards the identification of fake news. But in this proposed approach, author first trains an SVM with TF-IDF cosine similarity feature to determine whether a headline–article pair is related or unrelated. Then, the author builds various neural network architectures on top of Long-Short-Term-Memory (LSTM) to predict labels including only agree, disagree, or discuss pairing. Experiments report that a pair of Bidirectional Conditionally Encoded LSTM with Bidirectional GLobal Attention provides the best performance.

An approach consisting of three modules—Capture, Score, and Integrate (CSI)-based hybrid deep model, for fake news detection is proposed by Ruchansky et al. [55]. CSI is a multimodal approach considering three characteristics of a fake news. First one is the content of the article itself, then the behavior of user and article, and finally the group behavior of users who propagate the article. In this, the capture module uses the temporal representation of the article and a Recurrent Neural Network (RNN). The score is another module which learns the source characteristics based on the behavior of the user and finally, the integration module combines the previous two modules. Several other deep neural architectures for fake news detection are also reported in the literature [56–59].

Textual modality is one of the powerful ways of detecting whether an information is legitimate or fake. Other than the fake news detection, we can use textual modality to detect rumor [60], hoax [61], stance [62], satire [63], humor [64, 65], irony [66], deception [67], and misinformation in online reviews [68].

10.2.2 Image Modality

Images attract attention than text for being vivid and easily comprehensible. They are prevalent on social networking sites and reach a large number of people rapidly.

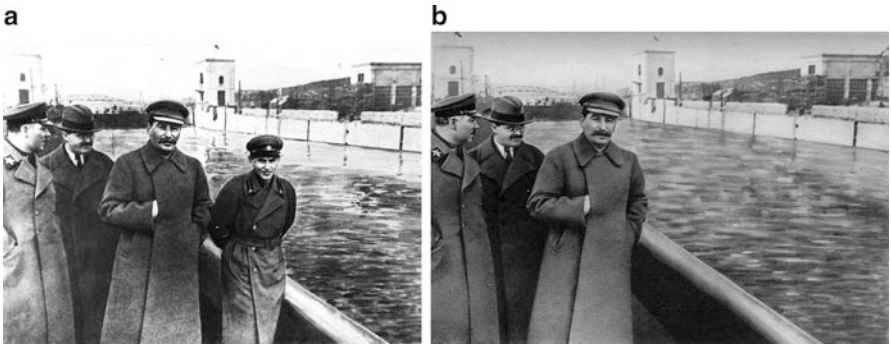


Fig. 10.3 The Vanishing Commissar. (**a**) Joseph Stalin pictured with Nikolai Yezhov. (**b**) Retouched photograph removing Nikolai Yezhov

But, these images propagating through the networking sites are prone to various forgeries to present false information to the users. The art of making fake images is not new and dates back to the twentieth century. For instance, the "Vanishing Commissar"[5] (Fig. 10.3a) where Joseph Stalin pictured with the Nikolai Yezhov was retouched to entirely remove Yezhov (Fig. 10.3b), from an official press photo for spreading propaganda.

Nowadays, there exist much software that allows even relatively inexperienced users to edit the digital images with such a perfection without leaving any trace of tampering, that it is hard to distinguish the forged images from the original photographs. There also exists powerful software that artificially generates highly photorealistic images, as if taken by a digital camera and deceive human eyes. And hence, we are no longer in a world where seeing is believing. This brings into account the significance of assessing the trustworthiness of an image propagating through the networking sites.

Fake images on social media can be broadly categorized as outdated images, inaccurate images, and manipulated images. *Outdated images* associate old images with a current news event. An example of outdated images is shown in Fig. 10.4a, which claims an image of two children in Nepal earthquake in 2015 but is originally an image of two Vietnamese taken in 2007.[6] *Inaccurate images* are used to describe false events. An example showing inaccurate images is given in Fig. 10.4b, where the image claims as a kind girl helping a homeless old man is a posed picture for commercial exploration.[7] Figure 10.4c shows a six-head snake as an example of manipulated or *tampered images*.[8]

[5] http://webarchive.nationalarchives.gov.uk/20120203152804/http://www.tate.org.uk/tateetc/issue8/erasurerevelation.htm.

[6] https://www.snopes.com/fact-check/nepal-earthquake-photo/.

[7] http://www.chinadaily.com.cn/china/2013-03/29/content_16357003.htm.

[8] https://www.snopes.com/fact-check/7-headed-snake/.

Fig. 10.4 Fake images on social media. (**a**) Outdated image. (**b**) Inaccurate image. (**c**) Tampered image

Image tampering can of three different types—image splicing, copy-move attack, and resampling/histogram operations. The method of copying one part of an image and inserting it into another refers to *image splicing* operation which is done to realize impossible facts. Image splicing detection algorithms employ the idea of finding out regions in an image that are significantly different from rest of the image [69–74]. The *copy-move attacks* practice taking a part of an image and copying it in the same image, to conceal something or to falsely add information. This type of attacks is detected using common image content search algorithms to find the internal replications [75–80]. The third type of tampering called *resampling/histogram operations* is well-intended and rarely change image semantics [81–85].

There are only a few works related to verifying the authenticity of a news article from the image modality aspect. Most of the works among them are based on the textual features surrounding the images. Others include user profile image features, temporal features of the image posted, etc. and very few are based on image forensic features.

A characterization analysis has been performed by Gupta et al. [16] to identify the fake images in Online Social Media and to understand the propagation of these images using temporal, social reputation, and influence patterns. In particular, the spread of fake images of Hurricane Sandy, during 2012, through Twitter has been highlighted in this paper. Figure 10.5 shows some of the fake images of Hurricane Sandy shared via twitter.[9] 10,350 unique tweets with fake images have been identified to be circulated on Twitter. For the classification analysis, a dataset containing 5767 fake and real image URLs each, of Hurricane Sandy, has been prepared, and classification algorithms have been used on the extracted user-based features like number of friends, followers, etc. and tweet-based features like length of tweet, number of words, etc., to distinguish the fake and real image URLs.

[9] http://news.yahoo.com/10-fake-photos-hurricane-sandy-075500934.html.

Fig. 10.5 Fake image tweets of Hurricane Sandy. (**a**) Shark supposedly swimming through Houston's flooded streets. (**b**) Thunderstorm at Manhattan

An accuracy of 97% has been achieved using decision tree classifier with tweet-based features implying that the analysis of tweet content and property helps to classify fake and real image URLs on Twitter with a high accuracy, but still, the work has been carried out using the traditional features based on text, user, and propagation and not the image features. Another interesting result has been inferred that approximately 86% of the tweets with fake images URLs are retweets.

From the conclusions of the abovementioned work of Gupta et al. [16], Jin et al. [86] comes up with the assumption that for a fake news there will be less diverse and limited number of images. The experiments are run on their real-world multimedia dataset prepared from Sina Weibo[10] constituting 146 news events, 50,287 tweets, 25,953 images, and their ground truths created from authoritative sources. Some of the real and fake images in their dataset are shown in Fig. 10.6, respectively. In this work, they explore the role of visual and statistical image features for automatic news verification on microblogs. They compute five visual features: Visual Clarity Score, Visual Coherence Score, Visual Similarity Distribution Histogram, Visual Diversity Score, and Visual Clustering Score. These features reveal the hidden characteristics of image distributions in a news event. Apart from visual features, they also define seven image statistical features from the three aspects, number of images in a tweet, popularity of images, and length-to-width ratio of images. These features provide information on the statistics and attributes of an image in a news

[10] http://service.account.weibo.com/.

Fig. 10.6 Sample images from Sina Weibo. (**a**) Real images. (**b**) Fake images

event. To compare the results of the visual and statistical image features, they extract the nonimage features from the aspects of text, propagation, and user proposed in [87–90] and remove the redundancy in the combination of these 42 nonimage features by selecting the 11 best features. These features train 4 classification models: Support Vector Machine, Logistic regression, KStar, and Random Forest with fourfold cross-validation. They infer that the performance of image features by combining their 5 extracted visual and 7 extracted statistical features is slightly inferior to the nonimage features and hence the effectiveness of nonimage features cannot be neglected.

A work to analyze online news has been proposed by Pasquini et al. in [91] that allows verification of the consistency of images in a news article with similar images of the same topic. This work has been proposed as a preliminary step towards verifying the integrity of the images which can be performed by visual inspection or by using multimedia forensics tools. For a webpage, the textual metadata has been extracted by identifying similarity towards the title and keywords of the news article. Similarly, the visual metadata has been extracted by Speeded Up Robust Features (SURF) feature matching, correlation value, and face detection using Maximally Stable Extremal Regions (MSER) feature matching. These image features have been used to identify some of the common cases in the news like same objects from different views or the images that are composite of people. For result analysis, a webpage with similar textual metadata and manipulated images have been created by the authors to observe the behavior of the approach towards a news with manipulated images. The system uses recursive retrieval

Fig. 10.7 An example image tweet used in the dataset of [92]

procedure and a number of links and their associated images have been retrieved for each news article. But, these images have been manually compared and classified based on their visual similarity, which may not be applicable for sophisticated and highly realistic manipulations and requires the employment of digital forensic techniques. The authors also suggest the application of the proposed work for detecting decontextualized images through a reverse approach by first retrieving similar images of a news and then comparing its textual information.

The paper by Hossain et al. analyzes the top trending images on social media to make sure that the images correspond to the claimed description and is real, through a collaborative analysis model [92]. An example image used in this study is shown in Fig. 10.7. The method searches social networks using tags describing an image to find out whether the image has been used in another context before. They also apply image forgery detection algorithm on the popular images retrieved using these tags. For forgery detection, the feature vectors are the statistical measures calculated from the co-occurrence matrix of the Steerable Pyramid Transform (SPT) subbands over the Chroma component of the image. This feature vector is then supplied to a Support Vector Machine (SVM) for classification.

The work by Jin et al. analyzes the image credibility by exploiting visual content to learn image representations using deep convolutional neural network (CNN) model [93]. They suggest the CNN model to capture complicated fake images' characteristics. They construct a large-scale auxiliary image set from fake tweets in social media consisting of 600,000 weakly labeled real and fake images. They use transfer learning to train the CNN model with the auxiliary and the target training set. They create the target image set consisting of 14,616 images with ground truth labels, from Sina Weibo and Xinhua News Agency for fake and real images, respectively. They find that their proposed method outperforms other baseline methods utilizing text-based features [16], Bag-of-Visual-Words (BoVW) features [94], ImageNet features [95], etc. They also perform an iterative transfer learning method which boosts the performance of the system by about 10% higher than the traditional text feature-based methods thus concluding the importance of the visual content in image credibility analysis.

The work by Markos et al. investigates state-of-the-art image splicing detection algorithms on the realistic Web images disseminated through social media [96]. They study the alterations like recompression and resampling produced on the images while passing through the social media and their impact on tampering detection. They present a real-world dataset, the Wild Web tampered image dataset along with the corresponding ground truths, and evaluate the state-of-the-art image splicing detection algorithms [97–101] on this dataset in real-world verification settings. They infer that most of these state-of-the-art image splicing detection algorithms are specialized for certain cases. For example, the work [99] seems to work better on the Columbia Uncompressed Image Splicing Detection Evaluation Dataset [102]. Since most of the image splicing detection algorithms failed on the real Web images, they conclude that the transformations that are applied to images while they pass through the social media diminish the successful detection of forgeries. This work thus exposes the vast gap in the state-of-the-art approaches for successful forgery detection in social media.

10.2.3 Network Modality

Apart from content-based approaches, network behavioral properties provide fake news detection measures using network information [88, 103, 104]. The network modality constructs features like clustering coefficient, degree, etc. from specific networks built using the information of users who publish related social media posts. Different networks that are studied in this modality are stance network, co-occurrence network, extended networks, friendship networks, and diffusion networks. A *stance network* is constructed with nodes as posts corresponding to a news and the weights over the edges indicate stance similarity. The *co-occurrence network* is constructed from the user engagement information, counting the relevant posts of users to the same news article. The following/followee network of users posting related tweets is named as *friendship networks*. The features considered in the friendship network are number of nodes, number of edges, number of nodes without an outgoing edge, the density of Largest Connected Component (LCC), number of edges of LCC, proportion of nodes without incoming edges, the proportion of nodes without outgoing edges, proportion of isolated nodes, etc. *Diffusion network* is an extension of the friendship network for tracking the propagation of news using nodes as users and edges as the information diffusion paths. The density of LCC, number of nodes of LCC, number of edges of LCC, the proportion of nodes without incoming edges, the proportion of nodes without outgoing edges, proportion of isolated nodes, etc. are the features considered in the diffusion network. *Extended network* considers features like number of nodes, number of edges, number of nodes without an incoming edge, number of nodes without an outgoing edge, the density of LCC, number of edges of LCC, etc.

Gupta et al. determine the role of Twitter network graph in retweet propagation of the fake image tweets [16]. For a user on Twitter, the social network is his follower

graph. Gupta et al. analyze the percentage of information diffusion through this follower graph of a user by finding the percentage of overlap between the retweet and follower graphs and infers that in cases of crisis, people propagate tweets or retweets what they find trending, irrespective of whether they follow the user or not.

Kwon et al. use the structural properties of the network along with user-based features, linguistic features, and temporal features for analyzing the spreading patterns of rumor tweets [105]. They consider three propagation structures, namely the extended network, friendship network, and the diffusion network to estimate network density, clustering coefficient, number of nodes, number of edges, information on nonreciprocal interactions of users, etc. Even though the diffusion network size is small, it is simple and effective to capture the flow of rumors among users. Their results show that the network features take long time periods for better prediction and so they suggest these features as expensive to use during the initial stages of classification of rumors.

Kwon et al. identify key structural and linguistic features for classification of rumors and non-rumors [88]. They extract 15 structural properties from the friendship network, the diffusion network, and the LCC of the friendship network. The classification is performed using a decision tree, support vector machine, and random forest classifiers. They come to an inference that structural features like the fraction of information flow from low- to high-degree nodes, the fraction of singletons, etc. have high predictive power which indicates that initial rumor conversations begin from relatively low or medium influence users and then reach a wide set of networks.

10.2.4 Temporal Modality

Temporal modality recognizes and detects fake news by modeling the sequence of the message, which carries the fake news. The message generates a temporal sequence by ignoring the structural information. This post sequence can be modeled based on the message itself, using statistical methods like Conditional Random Fields (CRF), or message cluster.

The work by Gupta et al. aims to characterize and identify the propagation of fake pictures on Twitter that created panic among the people during the Hurricane Sandy of 2012 [16]. A temporal analysis is performed on their dataset of 10,350 tweets with fake image URLs by plotting the number of tweets per hour for fake image URLs. This shows that it takes only 12 h for the fake image URLs to become viral. This sudden spike in their propagation is via retweets of very few users. The work finds that, out of the 10,350 tweets with fake images URLs, only 14% were unique, the rest were all retweets.

Kwon et al. contribute providing insight into the spreading patterns of rumor tweets over the time windows starting from the initial 3, 7, 14, and 28 to 56 days of circulation [105]. For each time window, they extract four set of features, namely user-based features, linguistic features, structural features of the network,

and temporal features. For rumor events, the temporal features like periodicity of the external shock, external shock periodicity offset, interaction periodicity offset, extracted from the daily time series of the tweets show multiple periodic spikes that lead to a cyclic trend, whereas non-rumor events have a single prominent spike at the initial circulation phase that decays quickly over time. The dataset consists of 72 rumor events built from two rumor archives, snopes.com and urbanlegends.about.com, and 58 non-rumor events built from news sites like from times.com, nytimes.com, and cnn.com. Random forest classification with threefold cross-validation shows that even though temporal features give poor differentiation between rumor and non-rumor events for shorter time windows, for a duration of 14 days and longer these features exhibit high prediction performance.

Kwon et al. identify rumor characteristics by analyzing the temporal, structural, and linguistic properties [88]. They propose a new model of periodic time series, as an extension to the SpikeM model [106], called the Periodic External Shocks (PES) model, that shows that rumor events have fluctuations over time. Some among the 11 temporal features extracted are starting time of breaking news, the strength of external shock at birth, the strength of interaction periodicity, interaction periodicity offset, the strength of external shock, the periodicity of the external shock, external shock periodicity offset, etc. They find that among all features periodicity of external shock has the highest power of prediction which supports the PES as a good model for detection of rumor.

Ruchansky et al. propose a model called CSI combining three aspects—text content of an article, user response received, and users promoting an event are combined for fake news prediction [55]. The model contains three modules: Capture, Score, and Integrate where Capture module uses a Recurrent Neural Network (RNN) to capture the user's temporal information for a given event. The user temporal response is captured using a Long Short-Term Memory (LSTM). The score module is for learning the source characteristic and the third module classifies an event as fake or real.

10.2.5 Knowledge-Base or Fact-Checking Approaches

The significant motivation of fake news is to inject false claims into the news content and ride the readers into the world of misinformation. So, a suitable and straightforward way of detecting fake news is to investigate the truthfulness of the information claimed by the news story. An example is the story of strange 7-foot-tall creature found in Sante Fe, Argentina [107]. This article also presented an eye-catching photograph of the strange animal for attracting attention to the story which made it difficult to characterize the article as fake for a naive user (Fig. 10.8a). The best way in such cases is to check the facts about the claim and understand the truthfulness, which is done by snopes.com fact-checkers [108]. Fact-checkers helped in unraveling the fact that the abovementioned 7-foot-tall creature was a fake and digitally manipulated image based on concept art "Harry Potter werewolf"

Fig. 10.8 Fake image stating a strange 7-foot-tall creature that was found in Sante Fe. (**a**) Fake image. (**b**) Harry Potter werewolf

(Fig. 10.8b). In general, this kind of approaches is termed as the fact-checking or knowledge-based approaches. The aim of a fact-checking or knowledge-based approach is to collect the relevant combination of facts regarding the claim of a story from several web sources and assign a truth value to the claim. The fact-checking can be further classified into two, as *the human-in-the-loop* approaches, where human beings take part the major role of fact-finding and *computational* approaches where automated network models will check the fact.

10.2.5.1 Human-in-the-Loop Approaches

These approaches are human-centric where a person or group analyzes the fact of an information. This approach can be further categorized as *expert-oriented* fact-checking where the fact-checker is a domain expert (e.g., Full Fact,[11] FactChecker,[12] and PolitiFact) and *crowdsourcing*-based fact-checking where the fact-checkers are normal people in the crowd.

Expert-Oriented Fact-Checking

The statistics provided by Duke Reporters' Lab[13] says that there is a large boom happening in the area of fact-checking to flow against the enormous creation and propagation of fake news. The number of fact-checkers around the world has tripled over the past 4 years, increasing from 44 to 149 since the Duke Reporters' Lab first began counting these projects in 2014, a 239% increase [109]. Forecasting this

[11] https://fullfact.org/.
[12] http://factchecker.in/.
[13] https://reporterslab.org/.

trend, Google developed Google News in 2016, consisting of the news articles and its related coverages [110]. According to Brandtzaeg et al., fact-checking service is divided into three categories based on their area of concern [111]. These are political and public statements (e.g., FactCheck, PolitiFact, The Washington Post, Fact-Checker, CNN Reality Check, and Full Fact), online rumors and hoaxes (e.g., Snopes, Hoax-Slayer, ThruthOrFiction HoaxBusters, and Viralgranskaren—Metro), and specific topics or controversies or particular conflicts or narrowly scoped issues or events (e.g., StopeFake, TruthBeTold, RefugeeCheck, Climate Feedback, and Brown Moses Blog). Other than these three categories International Fact-Checking Network (IFCN) at Poynter illustrates a large list of fact-checkers, and they are promoting them for their excellence. In the present scenario, the expert-oriented fact-checkers are more imperious, but they are tedious to use in the real-time applications due to their time complexity and cost.

Brandtzaeg et al. in his research study show trust and distrust in online fact-checking services. Specifically, the study focuses on Snopes, FactCheck.org, and StopFake to identify usefulness and trustworthiness of fact-checking. Positive and negative post or conversations on fact-checking service are collected from the social media and an analytical study is performed for the identification of usefulness and trustworthiness of fact-checking. There are also several research approaches in the previous literature reported about related fact-checkers and misinformation trackers that help the process of fact-checking [112, 113].

Crowdsourcing for Fact-Checking

Contradiction to the expert-based fact-checking, the crowdsourcing-based fact-checking employs the power of the crowd by giving chance to the common people for annotating a news as legitimate or not. By using the benefits of crowdsourcing, every news articles can be distributed to a group of common people through the web, giving them the privilege to annotate or simple ways to flag towards fake or real articles. This annotation or flag values can be then aggregated to generate an overall decision about the given article. To address hoaxes and fake news, in 2016 Facebook came up with such a flagging option, which allow each user to flag their story if it shows a fake content [114]. The user-flagged news articles will be reported to signatory organizations of Poynters International Fact-Checking Code of Principles. If the fact-checking organizations identify a story as fake, it will get flagged as disputed, and there will be a link to the corresponding article explaining the reason. Stories that have been disputed may also appear lower in News Feed. Fiskkit[14] is another such platform which leverages the benefits of crowdsourcing for better online discussions and annotates whether a specific part of a news article is true, false, fallacy, etc.

[14] http://fiskkit.com/.

There are also few similar research approaches reported in the literature towards leveraging the use of crowdsourcing [115, 116]. Tschiatschek et al. in a research study developed a novel algorithm called DETECTIVE, which performs Bayesian inference for detecting the fake news, and also they formalize the user's flagging accuracy for fake news detection to improve the flags in future [115].

10.2.5.2 Computational Fact-Checking

Automatic identification and assignment of truth value to an article is the prime objective of computational fact-checking. So to assign the truth value to a claim revealed by an article, we have to automatically extract all relevant facts from the available primary, secondary sources, and all other external sources including the open web and structured knowledge graph. The open web consists of references that can be compared with the claim propagated by the article, whereas the knowledge graph approach represents a linked open data as a structured network, like DBPedia, Ontology graphs, etc. So, in the knowledge graph approach, the claim will be inferred from the open data network designed with facts.

Ciampaglia et al. propose a method for computational fact-checking by using the shortest path between concept nodes under properly defined semantic proximity metrics on knowledge graphs [117]. Experiments in this study are conducted using a public knowledge graph extracted from Wikipedia, and the claims are in the categories of history, entertainment, geography, and biographical information. Another interesting, new dimension of the research proposed by You Wu et al. utilizes the art of query processing or question answering into computational fact-checking [118]. The previous literature shows several other relevant approaches towards computational fact-checking [119].

Table 10.1 shows the different research in the literature, for characterizing and detecting fake news using the machine learning models and several modalities including textual, image, network, and temporal.

10.3 Datasets

The web is a major source to store and exchange heterogeneous data like text, image, video, etc. So, the easiest way to build datasets for the study of fake news is to crawl the web sources. Two different approaches are possible. The first one is to crawl the social media post, tweets, or web pages using crawlers like Import.io, and then manually annotate them using domain experts or the freedom of crowd (crowdsourcing). In another approach, we can directly use the API's of fact-checkers and crawl the news article along with the class label given by a domain expert. To recognizing fake news using network or temporal patterns, most of the datasets are created using Twitter [87, 120] and Facebook [121, 122], and few of the works are reported using Weibo [89]. Several API's are available to collect the relevant article,

Table 10.1 Different approaches for characterization and detection of fake news

Paper	Modality	Learning technique
[38]	Textual	Supervised and deep learning
[47]	Textual	SVM with Sequential Minimal Optimization (SMO)
[33]	Textual	Supervised learning (Linear SVM)
[34]	Textual	Supervised learning (random forest generic classifier, and orientation-specific classifier)
[40]	Textual	Supervised learning (SVM)
[53]	Textual	Deep learning (CNN, and Bi-LSTM)
[54]	Textual	Supervised learning and deep learning (SVM, and LSTM)
[56]	Textual	Deep learning
[57]	Textual	Deep learning
[58]	Textual	Deep learning
[59]	Textual	Deep learning
[16]	Image	Supervised (Naive Bayes, J48 decision tree)
	Temporal	Fake image URL tweets per hour in twitter
	Network	Overlap between retweet and follower graph in twitter
[86]	Image	Supervised Leaning (SVM, LR, KStar, and random forest)
[91]	Image	Feature matching (SURF), and face detection (MSER)
[96]	Image	Statistical median Kolmogorov–Smirnov statistic
[92]	Image	SVM
[93]	Image	CNN
[105]	Temporal	Random forest classifier
	Network	
[88]	Temporal	Decision tree, random forest, and SVM
	Network	
	Textual	

user, and network related details from social media. Forged or tampered images for recognizing the truthfulness of a news are also easily collected using these API and crawlers from public web spaces. Some of the publically available datasets in the directions of text, image, network, and temporal are listed below.

10.3.1 LIAR[15]

LIAR is a fine-grained six-class-labeled textual dataset for fake news detection, which includes the truth ratings for the each labels pants-fire, false, barely true, half-true, mostly true, and true. 2.8 K human-labeled short statement textual data in this LIAR dataset is collected using POLITIFACT.COMs API. The label distribution for

[15] https://www.cs.ucsb.edu/~william/data/liar_dataset.zip.

the pants-fire case is 1050 and all other labels vary from 2063 to 2638. The size of the training set is 10,269, and the size of validation and testing sets are 1284 and 1283, respectively. The measure of average token per statement is 17.9 [38].

10.3.2 Fact-Checking-LTCSS 2014[16]

Andreas Vlachos et al. developed this dataset for fact-checking, which contains 221 fact-checked text statements and associated information like the speaker, the time it was made, the verdict assigned by the journalists, the link to the webpage containing the full analysis of the verdict, the sources used, and a judgment on whether we deem the statement as suitable for automated fact-checking [37]. The data statements are collected from two online fact-checking portals named politifact.com and Channel4, and they have assigned these 221 statements into fine-grained five labels or classes including True, Mostly True, Half True, Mostly False, and False.

10.3.3 BuzzFeedNews[17]

This repository contains news published on Facebook page from nine different news agencies over a week close to the 2016 US election, six from large hyperpartisan Facebook pages (three each from the right partisanship and from the left partisanship), and three from large mainstream political news pages (Politico, CNN politics, and ABC News Politics). All the posts are fact-checked and labeled as mostly true, a mixture of true and false, and mostly false. The total number of post in left, right, and mainstream are 471, 666, and 1145, respectively.

10.3.4 Getting Real About Fake News: Kaggle Dataset[18]

This Kaggle repository is only an initial step to understand and recognize fake news. It contains text (fake news) and related metadata scraped from 244 websites tagged as "bullshit" by the BS Detector[19] Chrome Extension by Daniel Sieradski. The BS Detector is developed for checking the truthfulness of online news articles giving the label outputs instead of human annotators. The total data in the repository represents 12,999 posts in total.

[16] https://sites.google.com/site/andreasvlachos/resources.

[17] https://github.com/BuzzFeedNews/2016-10-facebook-fact-check.

[18] https://www.kaggle.com/mrisdal/fake-news.

[19] https://github.com/selfagency/bs-detector.

10.3.5 CREDBANK[20]

CREDBANK is a large-scale social media corpus, crowdsourced from Twitter, including approximately 60 million tweets collected between the mid-October 2014 and end of February 2015 [123]. All the collected tweets are then broken down into related 100 news events, with each event assessed for credibilities by 30 annotators. This dataset provides a platform to study the textual, network, or social context of fake news.

10.3.6 Fake News Net[21]

This is an ongoing data collection repository for fake news studies in different dimensions like textual data and social context features collected from the network [103]. The repository includes two major components, the first one is the news content and associated details like the source (author or publisher) of the article, headline (short text that aims to catch the attention of reader), body text (details of news story), and image or video (important part of body content and visual cues to frame the story). The second one is the social context information of the article extracted from social network sites. The social context includes the information on the engagement of fake articles on Twitter. These are user profile, post-related information, and user information including follower and followee.

10.3.7 Fake News Challenge (FNC)-1[22]

FNC-1 is the dataset developed for fake news challenge, which explores how AI techniques can be useful to beat against fake news. The FNC-1 repository is not directly related to the process of fake news detection, even though this dataset can be used for the initial level study of fake news called stance detection. This initial level of stance detection can serve as a useful building block for fact-checking. This dataset consists of headline and body text, either from the same or two different news articles. The class labels for this headline–body text pair is classified into four: Agree, DIsagree, Discuss, and Unrelated.

[20] http://compsocial.github.io/CREDBANK-data/.

[21] https://github.com/KaiDMML/FakeNewsNet.

[22] https://github.com/FakeNewsChallenge/fnc-1.

10.3.8 Higgs Twitter Dataset[23]

Higgs dataset is the popular dataset among social media network analysis [124]. This dataset was created during and after the discovery of Higgs boson on 4th July 2012. In this dataset, a four-directional network statistics including social network statistics, retweet network statistics, reply network statistics, and mention network statistics are extracted from the Tweeter. The dataset was first used to study the anatomy of a scientific rumor, a kind of misinformation [124].

10.3.9 The Wild Web Tampered Image Dataset[24]

The Wild Web dataset contains realistic tampered web images disseminated through social media. This dataset was developed keeping in mind that most of the images found on the web are usually prone to post-processing operations like recompression and rescaling. Due to this fact, a vast majority of the tamper detection algorithms often perform poorly in the real world compared to the reported performance [96]. The Wild Web dataset contains 13,577 forged images (80 cases of forgeries) collected from various social media sources, along with the ground truth binary masks localizing the forgery.

10.3.10 CASIA Tampered Image Detection Evaluation Database[25]

CASIA has versions CASIA TIDEV1.0 and CASIA TIDEV2.0. The former version contains 800 authentic and 921 spliced color JPEG images, each of size 384,256 pixels. Most of the authentic images are collected from Corel image dataset and belong to several categories like scene images, animal images, plant images, nature images, etc. They create spliced images by crop-and-paste operation on authentic images. The later version V2.0 contains 7491 authentic and 5123 tampered color images of different sizes varying from 240,160 to 900,600 pixels. This version contains JPEG images with different Q factors and uncompressed image samples.

[23] https://snap.stanford.edu/data/higgs-twitter.html.

[24] http://mklab.iti.gr/project/wild-web-tampered-image-dataset.

[25] http://forensics.idealtest.org/.

10.3.11 Columbia Uncompressed Image Splicing Detection Evaluation Dataset[26]

This dataset contains 183 authentic images taken using one camera with EXIF information, and 180 spliced images created from the authentic images without post-processing. These are uncompressed images of size ranging from 757×568 to 1152×768 pixels in TIFF and BMP formats. Most of the images are indoor scenes [125].

10.4 Open Issues and Future Directions of Research

The study of characterizing and detecting fake news open up several related research areas and challenges. Stance detection is one among the related research area, which is effectively fit into the fact-checking and fake news detection pipeline [126, 127]. FCN-I introduce stance detection in the challenge stage-1 as a helpful initial step towards fake news identification and address how the article is related to its heading. Hoax detection [61], and humor detection [65] is also another such area that opens the path to fake news detection research. Classifying rumor is yet another pipeline for fake news detection. The general role of the rumor is to create and propagate ambiguous situations. In the previous studies, rumor analysis is divided into four subtasks including rumor detection, rumor tracking, stance classification, and veracity classification [128]. The traditional styles of truth detection and deception detection can be adapted as another approach for fake news detection in some other way [36, 129]. Despite these approaches, clickbait [130], spammer [131], and bot detection which are not directly related to this area of research are useful in fake news detection. Some fake news articles may be created with clickbait headlines to attract readers and get more revenue [132]. Spammers and bot detection research really aims to detect malicious users in the social and web space [133]. These spammer and social bots give the insights of malicious accounts that may aid the fake news detection.

Content-based fake news detection research comprises two important modalities, namely the textual and image modality. Among these two modalities, most of the works in the literature use textual modality-based approach to recognize and identify fake news. There are very few works reported based on image modality. The traditional approaches in textual modality-based fake news detection aim to use the machine learning models with several features extracted from the text corpus. In feature engineering, most of the works rely on the combination of linguistic, psycholinguistic, stylometric, statistical, syntactic, and semantic features. Among these features, few studies are proposed based on the sentiment or emotional

[26] http://www.ee.columbia.edu/ln/dvmm/downloads/authsplcuncmp/.

elements embedded in the news corpus. According to Vosoughi et al., fake news stories inspire fear, disgust, and surprise in replies, and real news stories inspire anticipation, sadness, joy, and trust [14]. The studies utilizing this fact that how does the presence of emotional parameters influence the characterization and detection of fake news is an area to be explored. Similar to emotional context of the study, the presence of sentiment imbibed unigrams is also not explored so well, but in some cases, researchers explore the total sentiment polarity (positive or negative), the presence of negation in the document, etc. along with other psycholinguistic features using LIWC [35]. Using LIWC, the psycholinguistic aspects of fake news are studied in several previous works of the literature but research dealing with the psychology context of fake news, for example, the intention detection of a particular fake news spread is not studied well.

Apart from the feature engineering, traditional machine learning models like SVM, Random Forest, Decision tree are explored a lot in several works in the previous research works. Ensemble learning approaches [134, 135], which use multiple learning models to obtain better predictive performance than could be obtained from the traditional models, can also be the other way to explore fake news detection. Transfer learning, in another way, is useful for the problem of learning and reusing knowledge from multiple different domains [136]. Using transfer learning, fake news characterization and detection problem can be learned from different feature space like raw and semantic features. It may improve the results from traditional learning models. Other than the ensemble and transfer learning, word embedding-based methods is another area to be mined. Deep learning methods using neural computing including the various models of RNN and LSTM is yet another direction of scope.

The exponential rise in the spread of communication has made fake news articles to follow three major properties: dynamicity (it can happen fast), deception (hard to verify), and homophily (consists with one's beliefs). Due to these three properties of fake news, it is hard to create a benchmark dataset with community agreed labels. Different types of fake news propagating through social media related to politics, health, disaster, etc. have different textual characteristics, but datasets are not yet created considering the domain features. Even though few datasets are available in the literature, they have only a few entries with which it is not possible to study the overall characteristics of fake news. Some datasets only consider the fake news detection as a binary classification problem, but the fact-checking platforms consider the fake news as a multi-class classification problem.

Visual information that attracts people to a news is a major component that is being forged nowadays. But, there are no much research studies reported in this area, most of them being related to the textual and temporal information corresponding to an image or video posted on the social network. But, since a large number of manipulations are being done on news images and video footages, there is a high requirement for a digital image or video forensic investigation in this area. The greatest challenge is a dataset of real and fake images or videos. Most of the available forged image datasets differ from the real-world set due to a series of resampling and recompression operations happening on them.

In case of network modality, the user–post relations is an area to explored for better predictions apart from user profile-based features that are reported in the literature. Another direction can be network embedding that embeds into a low-dimensional vector space [137]. Most of the propagation features of the network and temporal modality show poor prediction at the early stage of propagation of a fake news. Fake news detection studies can be extended to incorporate the ideas behind the human-in-the-loop or knowledge base approaches with other automated approaches. A multimodal approach towards fake news detection that combines several modalities and learning techniques for an accurate fake news detection is a challenge and necessity.

10.5 Conclusion

Understanding the fake news ecosystem is crucial even for the common people because it has an impact on the day-to-day events like health, social, cultural, political aspects, etc. The rate of propagation of fake news through online social media is exponential compared to that in traditional print media. Due to this trend in the propagation of fake news, the impact of fake news is unpredictable. We begin this chapter by introducing misinformation and its different forms. In this study, we consider fake news detection, in the perspective of different modalities and learning models. We give special emphasis on different fake news detection approaches in the literature through textual and image modality. Both textual and image modality approaches utilize the traditional feature-based machine learning techniques to the current deep learning techniques. In this study, we have observed that there is a large scope for creating a community agreed benchmark dataset in both textual and image modality-based fake news detection. We also perform a pilot study of the network and temporal modalities for fake news detection. Other than these modalities, this chapter reviews several knowledge-based or fact-checking approaches including the human-in-the-loop methods (crowdsourcing and fact-checking) and automated fact-checking. We also discuss the details and availability of different datasets for the future research and conclude the chapter by mentioning various open issues and future directions.

References

1. Palen, L., Anderson, K.M., Mark, G., Martin, J., Sicker, D., Palmer, M., Grunwald, D.: A vision for technology-mediated support for public participation & assistance in mass emergencies & disasters. In: Proceedings of the 2010 ACM-BCS Visions of Computer Science Conference, p. 8. British Computer Society, Swindon (2010)
2. Palen, L., Vieweg, S.: The emergence of online widescale interaction in unexpected events: assistance, alliance & retreat. In: Proceedings Conference on Computer Supported Cooperative Work, pp. 117–126. ACM, New York (2008)

3. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes twitter users: real-time event detection by social sensors. In: Proceedings of the 19th International Conference on World Wide Web, pp. 851–860. ACM, New York (2010)
4. Sakaki, T., Toriumi, F., Matsuo, Y.: Tweet trend analysis in an emergency situation. In: Proceedings of the Special Workshop on Internet and Disasters, p. 3. ACM, New York (2011)
5. Cheong, F., Cheong, C.: Social media data mining: a social network analysis of tweets during the 2010–2011 Australian floods. In: Proceedings of PACIS, vol. 11, pp. 46–46 (2011)
6. Verma, S., Vieweg, S., Corvey, W.J., Palen, L., Martin, J.H., Palmer, M., Schram, A., Anderson, K.M.: Natural language processing to the rescue? extracting" situational awareness" tweets during mass emergency. In: Proceedings of ICWSM, Barcelona, pp. 385–392 (2011)
7. Vieweg, S., Hughes, A.L., Starbird, K., Palen, L.: Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1079–1088. ACM, New York (2010)
8. Søe, S.O.: Algorithmic detection of misinformation and disinformation: Gricean perspectives. J. Doc. **74**(2), 309–332 (2018)
9. Krishna Kumar, K.P., Geethakumari, G.: Detecting misinformation in online social networks using cognitive psychology. Human-Centric Comput. Inf. Sci. **4**(1), 14 (2014)
10. Tandoc, E.C. Jr., Lim, Z.W., Ling, R.: Defining fake news. Digit. Journalism **6**(2), 137–153 (2018)
11. Gelfert, A.: Fake news: a definition. Informal Logic **38**(1), 84–117 (2018)
12. Weir, W.: History's greatest lies. Fair Winds, Beverly, MA (2009)
13. Dizikes, P.: http://news.mit.edu/2018/study-twitter-false-news-travels-faster-true-stories-0308. March 2018
14. Vosoughi, S., Roy, D., Aral, S.: The spread of true and false news online. Science **359**(6380), 1146–1151 (2018)
15. Willingham, A.J.: https://edition.cnn.com/2017/09/08/health/fake-images-posts-disaster-trnd/index.html. September 2017
16. Gupta, A., Lamba, H., Kumaraguru, P., Joshi, A.: Faking Sandy: characterizing and identifying fake images on twitter during Hurricane Sandy. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 729–736. ACM, New York (2013)
17. Mendoza, M., Poblete, B., Castillo, C.: Twitter under crisis: Can we trust what we RT? In: Proceedings of the First Workshop on Social Media Analytics, pp. 71–79. ACM, New York (2010)
18. Xiaochi, Z.: Internet rumors and intercultural ethics-a case study of panic-stricken rush for salt in China and iodine pill in America after Japanese earthquake and tsunami. Stud. Lit. Lang. **4**(2), 13 (2012)
19. Rapoza, K.: Can fake news impact the stock market? Forbes, 26 February 2017
20. Fernández-Luque, L., Bau, T.: Health and social media: perfect storm of information. Healthcare Inf. Res. **21**(2), 67–73 (2015)
21. Marcon, A.R., Murdoch, B., Caulfield, T.: Fake news portrayals of stem cells and stem cell research. Regen. Med. **12**(7), 765–775 (2017)
22. Starbird, K., Maddock, J., Orand, M., Achterman, P., Mason, R.M.: Rumors, false flags, and digital vigilantes: misinformation on twitter after the 2013 Boston marathon bombing. In: iConference 2014 Proceedings (2014)
23. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. J. Econ. Perspect. **31**(2), 211–236 (2017)
24. Jin, Z., Cao, J., Guo, H., Zhang, Y., Wang, Y., Luo, J.: Rumor detection on twitter pertaining to the 2016 US presidential election (2017). Preprint, arXiv:1701.06250
25. Shin, J., Jian, L., Driscoll, K., Bar, F.: Political rumoring on twitter during the 2012 US presidential election: rumor diffusion and correction. New Media Soc. **19**(8), 1214–1235 (2017)
26. Wilson, J.: Playing with politics: political fans and twitter faking in post-broadcast democracy. Convergence **17**(4), 445–461 (2011)

27. Giglietto, F., Iannelli, L., Rossi, L., Valeriani, A.: Fakes, news and the election: a new taxonomy for the study of misleading information within the hybrid media system (2016)
28. Guess, A., Nyhan, B., Reifler, J.: Selective exposure to misinformation: evidence from the consumption of fake news during the 2016 US presidential campaign (2018)
29. Kasprak, A.: https://www.snopes.com/fact-check/new-study-officially-declare-fluoride-neurotoxin/. April 2018
30. Evon, D.: https://www.snopes.com/fact-check/did-woman-infect-deliberately-hiv/. April 2018
31. Mikkelson, D.: https://www.snopes.com/fact-check/war-on-christmas-monument/. March 2018
32. Jacobson, L.: http://www.politifact.com/truth-o-meter/statements/2018/apr/19/donald-trump/donald-trump-correct-about-size-us-trade-deficit-j/. April 2018
33. Perez-Rosas, V., Kleinberg, B., Lefevre, A., Mihalcea, R.: Automatic detection of fake news (2017). Preprint, arXiv:1708.07104
34. Potthast, M., Kiesel, J., Reinartz, K., Bevendorff, J., Stein, B.: A stylometric inquiry into hyperpartisan and fake news (2017). Preprint, arXiv:1702.05638
35. Newman, M.L., Pennebaker, J.W., Berry, D.S., Richards, J.M.: Lying words: predicting deception from linguistic styles. Personal. Soc. Psychol. Bull. **29**(5), 665–675 (2003)
36. Feng, S., Banerjee, R., Choi, Y.: Syntactic stylometry for deception detection. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pp. 171–175. Association for Computational Linguistics, Stroudsburg (2012)
37. Vlachos, A., Riedel, S.: Fact checking: task definition and dataset construction. In: Proceedings of the ACL Workshop on Language Technologies and Computational Social Science, pp. 18–22 (2014)
38. Wang, W.Y.: "Liar, liar pants on fire": a new benchmark dataset for fake news detection (2017). Preprint, arXiv:1705.00648
39. Moschitti, A., Basili, R.: Complex linguistic features for text classification: a comprehensive study. In: European Conference on Information Retrieval, pp. 181–196. Springer, Berlin (2004)
40. Rubin, V., Conroy, N., Chen, Y., Cornwell, S.: Fake news or truth? Using satirical cues to detect potentially misleading news. In: Proceedings of the Second Workshop on Computational Approaches to Deception Detection, pp. 7–17 (2016)
41. Ott, M., Choi, Y., Cardie, C., Hancock, J.T.: Finding deceptive opinion spam by any stretch of the imagination. In: Proceedings of the 49th Annual Meeting: Human Language Technologies-Volume 1, pp. 309–319. ACL, Stroudsburg (2011)
42. Badaskar, S., Agarwal, S., Arora, S.: Identifying real or fake articles: towards better language modeling. In: Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II (2008)
43. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (2003)
44. Toma, C.L., Hancock, J.T.: Reading between the lines: linguistic cues to deception in online dating profiles. In: Proceedings of the Conference on Computer Supported Cooperative Work, pp. 5–8. ACM, New York (2010)
45. Pennebaker, J.W., Francis, M.E., Booth, R.J.: Linguistic inquiry and word count. Technical Report, Southern Methodist University, Dallas, TX (1993)
46. Ott, M., Cardie, C., Hancock, J.T.: Negative deceptive opinion spam. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 497–501 (2013)
47. Afroz, S., Brennan, M., Greenstadt, R.: Detecting hoaxes, frauds, and deception in writing style online. In: Symposium on Security and Privacy (SP), pp. 461–475. IEEE, Washington (2012)
48. Zheng, R., Li, J., Chen, H., Huang, Z.: A framework for authorship identification of online messages: Writing-style features and classification techniques. J. Assoc. Inf. Sci. Technol. **57**(3), 378–393 (2006)

49. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013). Preprint, arXiv:1301.3781
50. Goldberg, Y., Levy, O.: word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method (2014). Preprint, arXiv:1402.3722
51. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning, pp. 1188–1196 (2014)
52. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of the EMNLP, pp. 1532–1543 (2014)
53. Bhatt, G., Sharma, A., Sharma, S., Nagpal, A., Raman, B., Mittal, A.: On the benefit of combining neural, statistical and external features for fake news identification (2017). Preprint, arXiv:1712.03935
54. Chopra, S., Jain, S., Sholar, J.M.: Towards automatic identification of fake news: headline-article stance detection with LSTM attention models (2017)
55. Ruchansky, N., Seo, S., Liu, Y.: Csi: a hybrid deep model for fake news detection. In: Proceedings of the Conference on Information and Knowledge Management, pp. 797–806. ACM, New York (2017)
56. Chaudhry, Ali K., Baker, D., Thun-Hohenstein, P.: Stance detection for the fake news challenge: identifying textual relationships with deep neural nets. https://web.stanford.edu/class/cs224n/reports/2760230.pdf
57. Singhania, S., Fernandez, N., Rao, S.: 3HAN: a deep neural network for fake news detection. In: International Conference on Neural Information Processing, pp. 572–581. Springer, Berlin (2017)
58. Miller, K., Oswalt, A.: Fake news headline classification using neural networks with attention (2017)
59. Pfohl, S., Triebe, O., Legros, F.: Stance detection for the fake news challenge with attention and conditional encoding (2017)
60. Wu, L., Li, J., Hu, X., Liu, H.: Gleaning wisdom from the past: early detection of emerging rumors in social media. In: Proceedings of the International Conference on Data Mining, pp. 99–107. SIAM, Philadelphia (2017)
61. Vuković, M., Pripužić, K., Belani, H.: An intelligent automatic hoax detection system. In: International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, pp. 318–325. Springer, Berlin (2009)
62. Augenstein, I., Rocktäschel, T., Vlachos, A., Bontcheva, K.: Stance detection with bidirectional conditional encoding (2016). Preprint, arXiv:1606.05464
63. Burfoot, C., Baldwin, T.: Automatic satire detection: are you having a laugh? In: Proceedings of the IJCNLP Conference Short Papers, pp. 161–164. Association for Computational Linguistics, Stroudsburg (2009)
64. Mihalcea, R., Strapparava, C., Pulman, S.: Computational models for incongruity detection in humour. In: International Conference on Intelligent Text Processing and Computational Linguistics, pp. 364–374. Springer, Berlin (2010)
65. Mihalcea, R., Pulman, S.: Characterizing humour: an exploration of features in humorous texts. In: International Conference on Intelligent Text Processing and Computational Linguistics, pp. 337–347. Springer, Berlin (2007)
66. Reyes, A., Rosso, P.: On the difficulty of automatically detecting irony: beyond a simple case of negation. Knowl. Inf. Syst. **40**(3), 595–614 (2014)
67. Zhou, L., Burgoon, J.K., Nunamaker, J.F., Twitchell, D.: Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated communications. Group Decis. Negot. **13**(1), 81–106 (2004)
68. Mukherjee, A., Venkataraman, V., Liu, B., Glance, N.: Fake review detection: classification and analysis of real and pseudo reviews. Technical Report UIC-CS-2013–03, University of Illinois at Chicago (2013)
69. Meenakshi Sundaram, A., Nandini, C.: ASRD: algorithm for spliced region detection in digital image forensics. In: Computer Science On-line Conference, pp. 87–95. Springer, Berlin (2017)

70. Chen, W., Shi, Y.Q., Su, W.: Image splicing detection using 2-d phase congruency and statistical moments of characteristic function. In: Security, Steganography, and Watermarking of Multimedia Contents IX, vol. 6505, p. 65050R. International Society for Optics and Photonics, Leiden (2007)
71. He, Z., Sun, W., Lu, W., Lu, H.: Digital image splicing detection based on approximate run length. Pattern Recogn. Lett. **32**(12), 1591–1597 (2011)
72. Agarwal, S., Chand, S.: Image forgery detection using co-occurrence-based texture operator in frequency domain. In: Progress in Intelligent Computing Techniques: Theory, Practice, and Applications, pp. 117–122. Springer, Berlin (2018)
73. Abrahim, A.R., Rahim, M.S.M., Sulong, G.B.: Splicing image forgery identification based on artificial neural network approach and texture features. Clust. Comput. 1–14 (2018). https://doi.org/10.1007/s10586-017-1668-8
74. Dong, J., Wang, W., Tan, T., Shi, Y.Q.: Run-length and edge statistics based approach for image splicing detection. In: International Workshop on Digital Watermarking, pp. 76–87. Springer, Berlin (2008)
75. Li, J., Li, X., Yang, B., Sun, X.: Segmentation-based image copy-move forgery detection scheme. IEEE Trans. Inf. Forensics Secur. **10**(3), 507–518 (2015)
76. Thirunavukkarasu, V., Satheesh Kumar, J., Chae, G.S., Kishorkumar, J.: Non-intrusive forensic detection method using DSWT with reduced feature set for copy-move image tampering. Wirel. Pers. Commun. **98**(4), 3039–3057 (2018)
77. Huang, Y., Lu, W., Sun, W., Long, D.: Improved DCT-based detection of copy-move forgery in images. Forensic Sci. Int. **206**(1–3), 178–184 (2011)
78. Mahmood, T., Mehmood, Z., Shah, M., Saba, T.: A robust technique for copy-move forgery detection and localization in digital images via stationary wavelet and discrete cosine transform. J. Vis. Commun. Image Represent. **53**, 202–214 (2018)
79. Al-Qershi, O.M., Khoo, B.E.: Comparison of matching methods for copy-move image forgery detection. In: 9th International Conference on Robotic, Vision, Signal Processing and Power Applications, pp. 209–218. Springer, Berlin (2017)
80. Sunil, K., Jagan, D., Shaktidev, M.: DCT-PCA based method for copy-move forgery detection. In: ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India-Vol II, pp. 577–583. Springer, Berlin (2014)
81. Bunk, J., Bappy, J.H., Mohammed, T.M., Nataraj, L., Flenner, A., Manjunath, B.S., Chandrasekaran, S., Roy-Chowdhury, A.K., Peterson, L.: Detection and localization of image forgeries using resampling features and deep learning. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1881–1889. IEEE, Washington (2017)
82. Flenner, A., Peterson, L., Bunk, J., Mohammed, T.M., Nataraj, L., Manjunath, B.S.: Resampling forgery detection using deep learning and a-contrario analysis (2018). Preprint, arXiv:1803.01711
83. Choi, H.-Y., Hyun, D.-K., Choi, S., Lee, H.-K.: Enhanced resampling detection based on image correlation of 3d stereoscopic images. EURASIP J. Image Video Process. **2017**(1), 22 (2017)
84. Peng, A., Wu, Y., Kang, X.: Revealing traces of image resampling and resampling antiforensics. Adv. Multimedia **2017** (2017). https://doi.org/10.1155/2017/7130491
85. Popescu, A.C., Farid, H.: Exposing digital forgeries by detecting traces of resampling. IEEE Trans. Signal Process. **53**(2), 758–767 (2005)
86. Jin, Z., Cao, J., Zhang, Y., Zhou, J., Tian, Q.: Novel visual and statistical image features for microblogs news verification. IEEE Trans. Multimedia **19**(3), 598–608 (2017)
87. Castillo, C., Mendoza, M., Poblete, B.: Information credibility on twitter. In: Proceedings of the 20th International Conference on World Wide Web, pp. 675–684. ACM, New York (2011)
88. Kwon, S., Cha, M., Jung, K., Chen, W., Wang, Y.: Prominent features of rumor propagation in online social media. In: 2013 IEEE 13th International Conference on Data Mining (ICDM), pp. 1103–1108. IEEE, Washington (2013)

89. Wu, K., Yang, S., Zhu, K.Q.: False rumors detection on Sina Weibo by propagation structures. In: 2015 IEEE 31st International Conference on Data Engineering (ICDE), pp. 651–662. IEEE, Washington (2015)
90. Gupta, M., Zhao, P., Han, J.: Evaluating event credibility on twitter. In: Proceedings of the 2012 SIAM International Conference on Data Mining, pp. 153–164. SIAM, Philadelphia (2012)
91. Pasquini, C., Brunetta, C., Vinci, A.F., Conotter, V., Boato, G.: Towards the verification of image integrity in online news. In: 2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 1–6. IEEE, Washington (2015)
92. Hossain, M.S., Alhamid, M.F., Muhammad, G.: Collaborative analysis model for trending images on social networks. Futur. Gener. Comput. Syst. **86**, 855–862 (2017)
93. Jin, Z., Cao, J., Luo, J., Zhang, Y.: Image credibility analysis with effective domain transferred deep networks (2016). Preprint, arXiv:1611.05328
94. Zhang, S., Tian, Q., Hua, G., Huang, Q., Li, S.: Descriptive visual words and visual phrases for image applications. In: Proceedings of the 17th ACM International Conference on Multimedia, pp. 75–84. ACM, New York (2009)
95. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
96. Zampoglou, M., Papadopoulos, S., Kompatsiaris, Y.: Detecting image splicing in the wild (web). In: 2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 1–6. IEEE, Washington (2015)
97. Lin, Z., He, J., Tang, X., Tang, C.-K.: Fast, automatic and fine-grained tampered jpeg image detection via DCT coefficient analysis. Pattern Recogn. **42**(11), 2492–2501 (2009)
98. Bianchi, T., De Rosa, A., Piva, A.: Improved DCT coefficient analysis for forgery localization in JPEG images. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2444–2447. IEEE, Washington (2011)
99. Ferrara, P., Bianchi, T., De Rosa, A., Piva, A.: Image forgery localization via fine-grained analysis of CFA artifacts. IEEE Trans. Inf. Forensics Secur. **7**(5), 1566–1577 (2012)
100. Bianchi, T., Piva, A.: Image forgery localization via block-grained analysis of JPEG artifacts. IEEE Trans. Inf. Forensics Secur. **7**(3), 1003–1017 (2012)
101. Mahdian, B., Saic, S.: Using noise inconsistencies for blind image forensics. Image Vis. Comput. **27**(10), 1497–1503 (2009)
102. Hsu, Y.-F., Chang, S.-F.: Detecting image splicing using geometry invariants and camera characteristics consistency. In: 2006 IEEE International Conference on Multimedia and Expo, pp. 549–552. IEEE, Washington (2006)
103. Shu, K., Sliva, A., Wang, S., Tang, J., Liu, H.: Fake news detection on social media: a data mining perspective. ACM SIGKDD Explor. Newsl. **19**(1), 22–36 (2017)
104. Conroy, N.J., Rubin, V.L., Chen, Y.: Automatic deception detection: methods for finding fake news. Proc. Assoc. Inf. Sci. Technol. **52**(1), 1–4 (2015)
105. Kwon, S., Cha, M., Jung, K.: Rumor detection over varying time windows. PloS ONE **12**(1), e0168344 (2017)
106. Matsubara, Y., Sakurai, Y., Aditya Prakash, B., Li, L., Faloutsos, C.: Rise and fall patterns of information diffusion: model and implications. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 6–14. ACM, New York (2012)
107. Altered Dimensions. http://altereddimensions.net/2018/bizarre-7-foot-tall-creature-photographed-in-sante-fe-argentina. April 2018
108. Evon, D.: https://www.snopes.com/fact-check/mysterious-creature-terrorizing-argentina/?utm_source=socialflow&utm_medium=social. April 2018
109. Adair, B.: https://reporterslab.org/tag/international-fact-checking-network. June 2018
110. Gingras, R.: https://blog.google/topics/journalism-news/labeling-fact-check-articles-google-news/. October 2016

111. Brandtzaeg, P.B., Følstad, A.: Trust and distrust in online fact-checking services. Commun. ACM **60**(9), 65–71 (2017)
112. Guha, S.: Related fact checks: a tool for combating fake news (2017). Preprint, arXiv:1711.00715
113. Shao, C., Ciampaglia, G.L., Flammini, A., Menczer, F.: Hoaxy: a platform for tracking online misinformation. In: Proceedings of the 25th International Conference Companion on World Wide Web, pp. 745–750. International World Wide Web Conferences Steering Committee (2016)
114. Mosseri, A.: https://newsroom.fb.com/news/2016/12/news-feed-fyi-addressing-hoaxes-and-fake-news/. December 2016
115. Tschiatschek, S., Singla, A., Rodriguez, M.G., Merchant, A., Krause, A.: Detecting fake news in social networks via crowdsourcing (2017). Preprint, arXiv:1711.09025
116. Kim, J., Tabibian, B., Oh, A., Schölkopf, B., Gomez-Rodriguez, M.: Leveraging the crowd to detect and reduce the spread of fake news and misinformation. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 324–332. ACM, New York (2018)
117. Ciampaglia, G.L., Shiralkar, P., Rocha, L.M., Bollen, J., Menczer, F., Flammini, A.: Computational fact checking from knowledge networks. PloS ONE **10**(6), e0128193 (2015)
118. Wu, Y., Agarwal, P.K., Li, C., Yang, J., Yu, C.: Toward computational fact-checking. Proc. VLDB Endowment **7**(7), 589–600 (2014)
119. Magdy, A., Wanas, N.: Web-based statistical fact checking of textual documents. In: Proceedings of the 2nd International Workshop on Search and Mining User-Generated Contents, pp. 103–110. ACM, New York (2010)
120. Jin, F., Dougherty, E., Saraf, P., Cao, Y., Ramakrishnan, N.: Epidemiological modeling of news and rumors on twitter. In: Proceedings of the 7th Workshop on Social Network Mining and Analysis, p. 8. ACM, New York (2013)
121. Tambuscio, M., Ruffo, G., Flammini, A., Menczer, F.: Fact-checking effect on viral hoaxes: a model of misinformation spread in social networks. In: Proceedings of the 24th International Conference on World Wide Web, pp. 977–982. ACM, New York (2015)
122. Nguyen, N.P., Yan, G., Thai, M.T., Eidenbenz, S.: Containment of misinformation spread in online social networks. In: Proceedings of the 4th Annual ACM Web Science Conference, pp. 213–222. ACM, New York (2012)
123. Mitra, T., Gilbert, E.: Credbank: a large-scale social media corpus with associated credibility annotations. In: ICWSM, pp. 258–267 (2015)
124. De Domenico, M., Lima, A., Mougel, P., Musolesi, M.: The anatomy of a scientific rumor. Sci. Rep. **3**, 2980 (2013)
125. Hsu, Y.-F., Chang, S.-F.: Detecting image splicing using geometry invariants and camera characteristics consistency. In: International Conference on Multimedia and Expo (2006)
126. Zarrella, G., Marsh, A.: Mitre at semeval-2016 task 6: transfer learning for stance detection (2016). Preprint, arXiv:1606.03784
127. Mohammad, S., Kiritchenko, S., Sobhani, P., Zhu, X., Cherry, C.: Semeval-2016 task 6: detecting stance in tweets. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), pp. 31–41 (2016)
128. Zubiaga, A., Aker, A., Bontcheva, K., Liakata, M., Procter, R.: Detection and resolution of rumours in social media: a survey (2017). Preprint, arXiv:1704.00656
129. Li, Y., Gao, J., Meng, C., Li, Q., Su, L., Zhao, B., Fan, W., Han, J.: A survey on truth discovery. ACM Sigkdd Explor. Newsl. **17**(2), 1–16 (2016)
130. Potthast, M., Köpsel, S., Stein, B., Hagen, M.: Clickbait detection. In: European Conference on Information Retrieval, pp. 810–817. Springer, Berlin (2016)
131. Hu, X., Tang, J., Zhang, Y., Liu, H.: Social spammer detection in microblogging. In: Proceedings of IJCAI, vol. 13, pp. 2633–2639 (2013)
132. Chen, Y., Conroy, N.J., Rubin, V.L.: Misleading online content: recognizing clickbait as false news. In: Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection, pp. 15–19. ACM, New York (2015)

133. Ferrara, E., Varol, O., Davis, C., Menczer, F., Flammini, A.: The rise of social bots. Commun. ACM **59**(7), 96–104 (2016)
134. Opitz, D.W., Maclin, R.: Popular ensemble methods: an empirical study. J. Artif. Intell. Res. **11**, 169–198 (1999)
135. Polikar, R.: Ensemble based systems in decision making. IEEE Circuits Syst. Mag. **6**(3), 21–45 (2006)
136. Li, X., Rao, Y., Xie, H., Lau, R.Y.K., Yin, J., Wang, F.L.: Bootstrapping social emotion classification with semantically rich hybrid neural networks. IEEE Trans. Affect. Comput. **8**(4), 428–442 (2017)
137. Liao, L., He, X., Zhang, H., Chua, T.-S.: Attributed social network embedding (2017). Preprint, arXiv:1705.04969

Chapter 11
General Framework for Multi-View Metric Learning

Riikka Huusari, Hachem Kadri, and Cécile Capponi

Abstract We consider the problem of metric learning for multi-view data and present a general method for learning within-view as well as between-view metrics in vector-valued kernel spaces, as a way to capture multimodal structure of the data. We formulate a general convex optimization problem in this context to jointly learn the metric and the classifier or regressor in kernel feature spaces. The formulated multi-view metric learning (MVML) can be applied to data with any number of views, not just two, while as a kernel-based method it allows for various data types. Indeed, it is not required for the views to have the same data type, as long as all of them are individually kernelizable. We give concrete realizations of our iterative algorithm in both classification and regression settings, where the metric operating between views is also learned, either a full metric or a view-sparse one. In order to scale the computation to large training sets, a block-wise Nyström approximation of the multi-view kernel matrix is introduced. We justify our approach theoretically and experimentally, and show its performance on real-world datasets against relevant state-of-the-art methods.

11.1 Introduction

Multi-view learning refers to a learning framework where the data is described in multiple views (or modalities). For example, we might characterize birds with both their visual and audial characteristics, and would like to take both of these very different representations meaningfully into account in the learning model. The views available in the data might be not only correlated but also complementary,

This chapter is an expanded version of a previous conference paper [15].

R. Huusari (✉) · H. Kadri · C. Capponi
LIS, Aix-Marseille University, Toulon University, CNRS, Marseille, France
e-mail: Riikka.Huusari@lis-lab.fr; Hachem.Kadri@lis-lab.fr; cecile.capponi@lis-lab.fr

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_11

redundant, or contradictory, and it is important to consider these various view interactions in the learning process. Thus, learning over all the views is expected to produce a final classifier (or regressor) that is better than learning from each view individually. The goal of this work is to develop an algorithm that would learn these view interactions in a dataset and learn how to take advantage of them in solving the learning problem.

Multi-view learning is well known in the semi-supervised setting, where the agreement among views is usually optimized [4, 31]. Yet, the supervised setting has proven to be interesting as well, independently from any agreement condition on views. Co-regularization and multiple kernel learning (MKL) are two well-known kernel-based frameworks for learning in the presence of multiple views in data [36]. The former attempts to optimize measures of agreement and smoothness between the views over labeled and unlabeled examples [32]; the latter tries to efficiently combine multiple kernels defined on each view to exploit information coming from different representations [12].

Kernel methods are well known and much used learning paradigm in machine learning mainly because of their appealing theoretical properties [14]. Main motivation for using kernels in machine learning problems is the kernel trick, which allows inexpensive computation of inner products in potentially infinite-dimensional feature spaces. That is, the data is mapped to a higher-dimensional feature space but all computations can be done with the data in original space with a kernel function, and explicit calculations in feature spaces are never performed. Using kernels in linear classifiers, such as SVM, we will essentially be able to perform nonlinear classification by considering the linear method in the feature space.

Matrix-valued kernels (or operator-valued kernels) are a generalization of scalar-valued kernels. Intuitive difference between the two is that instead of a scalar, the kernel function in more general case outputs a matrix. The main advantage of matrix-valued kernels is that they offer a higher degree of flexibility in encoding similarities between data points. They have been applied with success in various machine learning problems, such as multitask learning [11], functional regression [17], and structured output prediction [6]. However, finding the optimal matrix-valued kernel of choice for a given application is difficult, as is the question of how to build them.

In some areas of research, such as in computational biology as well as computer vision, kernel-based methods in multi-view context are widely used [20, 27]. As already mentioned, one of the simplest and most known of the kernel-based multi-view learning methods is multiple kernel learning (MKL) framework [36], where a simple linear combination of scalar-valued kernel matrices calculated with data from each individual view is learned. More recently, vector-valued reproducing kernel Hilbert spaces (RKHSs) have been introduced to the field of multi-view learning for going further than MKL by incorporating in the learning model both within-view and between-view dependencies [16, 24]. It turns out that these kernels and their associated vector-valued reproducing Hilbert spaces provide a unifying framework for a number of previous multi-view kernel methods, such as

co-regularized multi-view learning and manifold regularization, and naturally allow to encode within-view as well as between-view similarities [25].

In order to overcome the need for choosing a kernel before the learning process, we propose a supervised metric learning approach that learns a matrix-valued multi-view kernel jointly with the decision function. We refer the reader to [3] for a review of metric learning. It is worth mentioning that algorithms for learning matrix-valued kernels have been proposed in the literature, see, for example, [9, 10, 21]. However, these methods mainly consider separable kernels which are not suited for multi-view setting, as will be illustrated later in this paper.

This chapter presents a general multi-view learning method that

- is suitable for data with any number of views,
- can be used with any type of kernelizable data (that is not restricted to be of the same type across the views),
- considers view interactions in learning process and learns a metric that operates between views.

This method, called multi-view metric learning (MVML), was originally introduced in [15]. There the focus has only been on a squared loss function in optimization, but here we consider a more general setting and offer also the solution in support vector machine context. The interest of this chapter is mainly on supervised multi-view learning, although a new, possible extension to semi-supervised setting is introduced. We also provide more thorough experiments by adding a new dataset and a competing method in experiments in classification context.

The chapter is organized as follows. In Sect. 11.2, we focus on theory of operator-valued kernels and how they are used in (multi-view) learning. Section 11.3 presents the MVML framework and introduces concrete realizations of the algorithm in both classification and regression context, as well as theoretical results obtained. We validate the presented method with experiments in Sect. 11.4.

11.2 Scalar- and Operator-Valued Kernels

This section is dedicated to the theory of vector-valued RKHSs and their associated operator-valued (or matrix-valued) kernels, and how these are used in (multi-view) learning. We consider these subjects only in extent needed to understand the multi-view metric learning framework. For further reading on matrix-valued reproducing kernels, see, e.g., [1, 7, 8, 17].

Notation Throughout the chapter, we use n as the number of labeled data samples and v as the number of views present in the data. We denote vectors as bold lowercase symbols ($\mathbf{a}$), and matrices as bold uppercase ($\mathbf{A}$). As we consider block-structured matrices, the subscripts such as $\mathbf{A}_{ij}$ refer to a block, not just one value. Data is denoted $x \in \mathcal{X}$ as we consider all types of data, not just those that have vector representation. We denote k as the traditional scalar-valued kernel function,

and K as operator-valued kernel function. Correspondingly, the kernel matrices are written as $\mathbf{K}$ and $\mathbf{G}$, and the RKHS's are $\mathscr{K}$ and $\mathscr{H}$. If $\mathscr{B}$ is a Hilbert space, we denote $\mathscr{L}(\mathscr{B})$ the set of bounded linear operators from $\mathscr{B}$ to $\mathscr{B}$.

11.2.1 Scalar-Valued RKHSs

Before going into detail about vector-valued RKHSs, let us recall a few facts about scalar-valued kernels. This section is not meant to be a comprehensive introduction to kernel methods, rather it should provide an easy way for comparing the matrix-valued kernels to scalar-valued ones. For a more thorough treatment, see, for example, [14].

The goal of a (supervised inductive) scalar-valued learning problem is to learn a function $f : \mathscr{X} \to \mathbb{R}$ (or to some subset of $\mathbb{R}$, in classification usually $\{-1, 1\}$). A scalar-valued kernel is a positive-definite function $k : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$, positivity meaning that for any $\alpha_i, \alpha_j \in \mathbb{R}$

$$\sum_{i,j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \geq 0$$

holds. An important theorem in kernel theory, representer theorem, states that for a large class of optimization problems we can represent f with kernel as:

$$f(\cdot) = \sum_{i=1}^{n} k(\cdot, x_i)\alpha_i,$$

where $\alpha_i \in \mathbb{R}$. The reproducing property of kernels is written as:

$$\langle k(\cdot, x), f \rangle_{\mathscr{K}} = f(x).$$

The motivation for using kernels in machine learning lies in the kernel trick; that is, the kernel function corresponds to an inner product in some feature space, $k(x, z) = \langle \phi(x), \phi(z) \rangle$. Basically, this is a way to map the data into some, possibly infinite-dimensional, feature space and to perform linear classification task there.

11.2.2 Vector-Valued RKHSs

This section presents general theory of vector-valued RKHSs, without considering the restriction to multi-view setting. We assume general type of data $x \in \mathscr{X}$ and general labels $y \in \mathscr{Y}$. For example, in the application of operator-valued kernels to multitask learning the label space $\mathscr{Y}$ equals to $\mathbb{R}^t$ where t is the number of tasks.

Vector-valued RKHSs were introduced to the machine learning community by Micchelli and Pontil [23] as a way to extend kernel machines from scalar to vector outputs, with the goal of learning nonlinear vector-valued functions.

Definition 1 (Vector-Valued RKHS) A Hilbert space $\mathscr{H}$ of functions from $\mathscr{X}$ to $\mathscr{Y}$ is called a vector-valued reproducing kernel Hilbert space if there is a positive definite $\mathscr{L}(\mathscr{Y})$-valued kernel K on $\mathscr{X} \times \mathscr{X}$ such that:

i. the function $z \mapsto K(x, z)y$ belongs to $\mathscr{H}$, $\forall\, z, y \in \mathscr{X}$, $y \in \mathscr{Y}$,
ii. $\forall\, f \in \mathscr{H}, x \in \mathscr{X}$, $y \in \mathscr{Y}$,

$$\langle f, K(x, \cdot)y\rangle_{\mathscr{H}} = \langle f(x), y\rangle_{\mathscr{Y}} \quad \text{(reproducing property)}.$$

Definition 2 (Matrix- or Operator-Valued Kernel) An $\mathscr{L}(\mathscr{Y})$-valued kernel K on $\mathscr{X} \times \mathscr{X}$ is a function $K(\cdot, \cdot) : \mathscr{X} \times \mathscr{X} \to \mathscr{L}(\mathscr{Y})$; it is positive semidefinite if:

i. $K(x, z) = K(z, x)^*$, where * denotes the transpose of a matrix (or adjoint of an operator),
ii. and, for every $r \in \mathbb{N}$ and all $\{(x_i, y_i)_{i=1,\ldots,r}\} \in \mathscr{X} \times \mathscr{Y}$,

$$\sum_{i,j} \langle y_i, K(x_i, x_j)y_j\rangle_{\mathscr{Y}} \geq 0.$$

If the output space $\mathscr{Y}$ is $\mathbb{R}^v$, then $\mathscr{L}(\mathscr{Y}) = \mathbb{R}^{v\times v}$. The kernel matrix, given n data samples, would in this case be of size $nv \times nv$, while the scalar-valued kernel matrix is only $n \times n$.

In this setting, given a random training sample $\{x_i, y_i\}_{i=1}^n$ on $\mathscr{X} \times \mathscr{Y}$, optimization problem:

$$\arg\min_{f\in\mathscr{H}} \sum_{i=1}^{n} V(f, x_i, y_i) + \lambda \|f\|^2_{\mathscr{H}}, \tag{11.1}$$

where f is a vector-valued function and V is a loss function, can be solved in a vector-valued RKHS $\mathscr{H}$ by the means of a vector-valued extension of the representer theorem.

Theorem 1 (Bijection Between Vector-Valued RKHS and Matrix-Valued Kernel) *An $\mathscr{L}(\mathscr{Y})$-valued kernel K on $\mathscr{X} \times \mathscr{X}$ is the reproducing kernel of some Hilbert space $\mathscr{H}$, if and only if it is positive semidefinite.*

Theorem 2 (Representer Theorem) *Let K be a positive semidefinite matrix-valued kernel and $\mathscr{H}$ its corresponding vector-valued RKHS. The solution $\hat{f} \in \mathscr{H}$ of the regularized optimization problem (11.1) has the following form:*

$$\hat{f}(x) = \sum_{i=1}^{n} K(x, x_i)c_i, \quad \textit{with} \quad c_i \in \mathscr{Y}. \tag{11.2}$$

With regard to the classical representer theorem, here the kernel K outputs a matrix and the "weights" c_i are vectors. The proof of Theorem 1 can be found in [17], and of Theorem 2 in [23]. For further reading on matrix-valued kernels and their associated RKHSs, see, e.g., [7, 8].

Examples of Operator-Valued Kernels Some well-known classes of matrix-valued kernels include separable and transformable kernels [1].

Definition 3 (Separable Kernel) An operator-valued kernel is called separable, if it can be written as:

$$K(x, z) = k(x, z)T,$$

where k is a scalar-valued kernel and $T \in \mathscr{L}(\mathscr{Y})$.

In multi-view setting, the matrix $\mathbf{T}$ is in $\mathbb{R}^{v \times v}$ where v is the number of views. This class of kernels is very attractive in terms of computational time, as it is easily decomposable. However, the matrix $\mathbf{T}$ acts only on the outputs independently of the input data, which makes it difficult for these kernels to encode necessary similarities in multi-view setting.

Definition 4 (Transformable Kernel) An operator-valued kernel is called transformable, if it can be written as:

$$[K(x, z)]_{lm} = k(S_m x, S_l z).$$

Here, m and l are indices of the output matrix (views in multi-view setting) and operators, $\{S_t\}_{t=1}^{v}$, are used to transform the data.

In contrast to separable kernels, here the S_t operate on input data; however, choosing them is a difficult task.

11.2.3 Vector-Valued Multi-View Learning

This section reviews the setup for supervised multi-view learning in vector-valued RKHSs [16, 25]. The main idea is to consider a kernel that measures not only the similarities between examples of the same view but also those coming from different views. Reproducing kernels of vector-valued Hilbert spaces allow encoding in a natural way these similarities and taking into account both within-view and between-view dependencies. Indeed, a kernel function K in this setting outputs a matrix in $\mathbb{R}^{v \times v}$, with v the number of views, so that $K(x_i, x_j)_{lm}, l, m = 1, \ldots, v$, is the similarity measure between examples x_i and x_j from the views l and m.

More formally, consider a set of n labeled data $\{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y}, i = 1, \ldots, n\}$, with $\mathscr{Y} = \{-1, 1\}$ for classification or $\mathscr{Y} \subset \mathbb{R}$ for regression. Assume that each input instance $x_i = (x_i^1, \ldots, x_i^v)$ is seen in v views. We note that the views are not

restricted to be of the same type. The supervised multi-view learning problem can be thought of as trying to find the vector-valued function $\hat{f}(\cdot) = (\hat{f}^1(\cdot), \ldots \hat{f}^v(\cdot))$, with $\hat{f}^l(x) \in \mathscr{Y}$, solution of

$$\underset{f \in \mathscr{H}, \mathscr{W}}{\arg\min} \sum_{i=1}^{n} V(y_i, \mathscr{W}(f(x_i))) + \lambda \|f\|^2. \tag{11.3}$$

Here, f is a vector-valued function that groups v learning functions, each corresponding to one view, and $\mathscr{W} : \mathbb{R}^v \to \mathbb{R}$ is combination operator for combining the results of the learning functions.

While the vector-valued extension of the representer theorem provides an algorithmic way for computing the solution of the multi-view learning problem (11.3), the question of choosing the multi-view kernel K remains crucial to take full advantage of the vector-valued learning framework. In [16], a matrix-valued kernel based on cross-covariance operators on RKHS that allow modeling variables of multiple types and modalities was proposed. However, it has two major drawbacks: (1) the kernel is fixed in advance and does not depend on the learning problem, and (2) it is computationally expensive and becomes infeasible when the problem size is very large. We avoid both of these issues by learning a block low-rank metric in kernel feature spaces.

11.3 Multi-View Metric Learning

Here, we introduce a general framework for learning simultaneously a vector-valued multi-view function and a positive semidefinite metric between kernel feature maps, as well as an operator for combining the answers from the views to yield the final decision. We then show concrete realizations of this framework in both classification and regression context, and for learning full or sparse metric matrices that operate between the views. We also give a Rademacher bound [2] for our algorithm(s). Finally, we demonstrate how it can be implemented efficiently via block-wise Nyström approximation [35].

11.3.1 Matrix-Valued Multi-View Kernel

We consider the following class of matrix-valued kernels that can operate over multiple views:

$$K(x_i, x_j)_{lm} = \left\langle \Phi_l\left(x_i^l\right), C_{\mathscr{X}_l \mathscr{X}_m} \Phi_m\left(x_j^m\right) \right\rangle, \tag{11.4}$$

where Φ_l (resp., Φ_m) is the feature map associated with the scalar-valued kernel k_l (resp., k_m) defined on the view l (resp., m). In the following, we will leave out the view label from data instance when the feature map or kernel function already has that information, e.g., instead of $\Phi_l(x_i^l)$ we write $\Phi_l(x_i)$. $C_{\mathscr{X}_l\mathscr{X}_m} : \mathscr{H}_m \to \mathscr{H}_l$ is a linear operator between the scalar-valued RKHSs $\mathscr{H}_l$ and $\mathscr{H}_m$ of kernels k_l and k_m, respectively. The operator $C_{\mathscr{X}_l\mathscr{X}_m}$ allows one to encode both within-view and between-view similarities.

The choice of the operator $C_{\mathscr{X}_l\mathscr{X}_m}$ is crucial and depends on the multi-view problem at hand. In the following, we only consider operators $C_{\mathscr{X}_l\mathscr{X}_m}$ that can be written as $C_{\mathscr{X}_l\mathscr{X}_m} = \boldsymbol{\Phi}_l \mathbf{A}_{lm} \boldsymbol{\Phi}_m^T$, where $\boldsymbol{\Phi}_s = (\Phi_s(x_1), \ldots, \Phi_s(x_n))$ with $s = l, m$, and $\mathbf{A}_{lm} \in \mathbb{R}^{n\times n}$ is a positive definite matrix which plays the role of a metric between the two feature maps associated with kernels k_l and k_m defined over the views l and m. This is a large set of possible operators but depends on a finite number of parameters. It gives us the following class of kernels:

$$\begin{aligned} K(x_i, x_j)_{lm} &= \left\langle \Phi_l(x_i), \boldsymbol{\Phi}_l \mathbf{A}_{lm} \boldsymbol{\Phi}_m^T \Phi_m(x_j) \right\rangle \\ &= \left\langle \boldsymbol{\Phi}_l^T \Phi_l(x_i), \mathbf{A}_{lm} \boldsymbol{\Phi}_m^T \Phi_m(x_j) \right\rangle \\ &= \langle \mathbf{k}_l(x_i), \mathbf{A}_{lm} \mathbf{k}_m(x_i) \rangle \,, \end{aligned} \tag{11.5}$$

where we have written $\mathbf{k}_l(x_i) = (k_l(x_t, x_i))_{t=1}^n$. We note that this class is not in general separable or transformable. However in the special case when it is possible to write $\mathbf{A}_{ml} = \mathbf{A}_m \mathbf{A}_l$, the kernel is transformable.

It is easy to see that the lm-th block of the block kernel matrix $\mathbf{G}$ built from the matrix-valued kernel (11.5) can be written as $\mathbf{G}_{lm} = \mathbf{K}_l \mathbf{A}_{lm} \mathbf{K}_m$, where $\mathbf{K}_s = \left(k_s(x_i, x_j)\right)_{i,j=1}^n$ for view s. The block kernel matrix $\mathbf{G} = \left(K(x_i, x_j)\right)_{i,j=1}^n$ in this case has the form:

$$\mathbf{G} = \mathbf{HAH}, \tag{11.6}$$

where $\mathbf{H} = \text{blockdiag}(\mathbf{K}_1, \ldots, \mathbf{K}_v)$,[1] and the matrix $\mathbf{A} = (\mathbf{A}_{lm})_{l,m=1}^v \in \mathbb{R}^{nv\times nv}$ encodes pairwise similarities between all the views. Multi-view metric learning then corresponds to simultaneously learning the metric $\mathbf{A}$ and the classifier or regressor.

From this framework, with suitable choices of $\mathbf{A}$, we can recover the cross-covariance multi-view kernel of [16], or, for example, an MKL-like multi-view kernel containing only one-view kernels.

[1] Given a set of $n \times n$ matrices $\mathbf{K}_1, \ldots, \mathbf{K}_v$, $\mathbf{H} = \text{blockdiag}(\mathbf{K}_1, \ldots, \mathbf{K}_v)$ is the block-diagonal matrix satisfying $\mathbf{H}_{l,l} = \mathbf{K}_l, \forall l = 1, \ldots, v$.

11.3.2 Learning Problem

Using the vector-valued representer theorem (Theorem 2), the multi-view learning problem (11.3) becomes

$$\underset{\mathbf{c}_1,\ldots,\mathbf{c}_n\in\mathbb{R}^v}{\arg\min} \sum_{i=1}^{n} V\left(y_i, \mathscr{W}\left(\sum_{j=1}^{n} K(x_i,x_j)\mathbf{c}_j\right)\right) + \lambda \sum_{i,j=1}^{n} \left\langle \mathbf{c}_i, K(x_i,x_j)\mathbf{c}_j \right\rangle. \tag{11.7}$$

We will assume that the operator $\mathscr{W}$ calculates a weighted sum of outputs and stores the weights in vector $\mathbf{w}$. In order to jointly learn the metric matrix $\mathbf{A}$ with the learning problem, we need to add additional regularizer for it. We write $\mathbf{H}_{x_i}$ to select the rows in block-matrix $\mathbf{H}$ that correspond to data sample x_i. Thus, we will write more precisely

$$\underset{\mathbf{c}\in\mathbb{R}^{vn}}{\arg\min} \sum_{i=1}^{n} V\left(y_i, \mathbf{w}^\top \mathbf{H}_{x_i}\mathbf{A}\mathbf{H}\mathbf{c}\right) + \lambda\langle \mathbf{c}, \mathbf{H}\mathbf{A}\mathbf{H}\mathbf{c}\rangle + \eta\Omega(\mathbf{A}), \tag{11.8}$$

where $\mathbf{A}$ should be a positive (semi)definite matrix. However, we do not implicitly write this into our optimization problem, since it turns out that we can optimize without this constraint, and the result will be positive.

In order to make the problem more easily solvable and convex, we introduce a change of variables $\mathbf{g} = \mathbf{A}\mathbf{H}\mathbf{c}$, and a mapping $(\mathbf{c}, \mathbf{A}) \rightarrow (\mathbf{g}, \mathbf{A})$. This is a valid change (see, e.g., [5]). The problem is now

$$\underset{\mathbf{g}\in\mathbb{R}^{vn}}{\arg\min} \sum_{i=1}^{n} V\left(y_i, \mathbf{w}^\top \mathbf{H}_{x_i}\mathbf{g}\right) + \lambda\langle \mathbf{g}, \mathbf{A}^\dagger\mathbf{g}\rangle + \eta\Omega(\mathbf{A}), \tag{11.9}$$

where † denotes pseudo-inverse. For the convexity of the problem, the main idea is to note that $\langle \mathbf{g}, \mathbf{A}^\dagger\mathbf{g}\rangle$ is jointly convex (see, e.g., [37]).

It is good to note that despite the misleading similarities between our work and that of [9], we use different mappings for solving our problems, which are also formulated differently. We also consider different classes of kernels as [9] considers only separable kernels.

We use an alternating scheme to solve our problem. Our MVML algorithm thus iterates over solving $\mathbf{A}$ and $\mathbf{g}$ and also $\mathbf{w}$ if weights are to be learned. In the following after the illustration, we introduce the solutions for this problem with two loss functions and two regularizers Ω in later sections.

11.3.3 Solving for A

We note from (11.9) that the metric matrix $\mathbf{A}$ is only present in the two regularization terms, $\lambda\langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g}\rangle$ and $\eta\Omega(\mathbf{A})$, thus we can solve for it independently from considering the framework we would be applying the method to.

We consider two choices of $\Omega(\mathbf{A})$. The first possibility is to regularize the complexity of the whole metric matrix $\mathbf{A}$ in Frobenius norm:

$$\Omega_1(\mathbf{A}) = \|\mathbf{A}\|_F^2 \tag{11.10}$$

The solution of (11.9) for $\mathbf{A}$ with fixed $\mathbf{g}$ is obtained by gradient descent, where the update rule is given by:

$$\mathbf{A}^{k+1} = (1 - 2\mu\eta)\,\mathbf{A}^k + \mu\lambda \left(\mathbf{A}^k\right)^\dagger \mathbf{g}\mathbf{g}^T \left(\mathbf{A}^k\right)^\dagger, \tag{11.11}$$

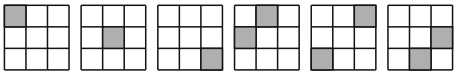
where μ is the step size.

As another possibility for regularization over the metric matrix, we consider sparsity on a group level so that whole blocks corresponding to pairs of views are put to zero. Intuitively, the block-sparse result will give insight as to which views are interesting and worth taking into account in learning. For example, by tuning the parameter controlling sparsity level one could derive, in some sense, an order of importance to the views and their combinations.

The regularization term in this case is

$$\Omega_2(\mathbf{A}) = \sum_{\gamma \in \mathscr{G}} \|\mathbf{A}_\gamma\|_F^2. \tag{11.12}$$

This is an l_1/l_2-regularizer over set of groups $\mathscr{G}$ we consider for sparsity. In our multi-view setting, these groups correspond to combinations of views; for example, with three views the matrix $\mathbf{A}$ would consist of six groups:



When we speak of combinations of views, we include both blocks of the matrix that this combination corresponds to. Using this group regularization, in essence, allows us to have view sparsity in our multi-view kernel matrix.

Using this sparsity-promoting regularization, the problem is solved w.r.t. $\mathbf{A}$ with proximal gradient method, the update rule being

$$[\mathbf{A}^{k+1}]_\gamma = \left(1 - \frac{\eta}{\|[\mathbf{A}^k - \mu^k \nabla h(\mathbf{A}^k)]_\gamma\|_F}\right)_+ [\mathbf{A}^k - \mu^k \nabla h(\mathbf{A}^k)]_\gamma, \tag{11.13}$$

where μ^k is the step size,

$$h(\mathbf{A}^k) = \lambda \left\langle \mathbf{g}, (\mathbf{A}^k)^\dagger \mathbf{g} \right\rangle \qquad \text{and}$$
$$\nabla h(\mathbf{A}^k) = -\lambda (\mathbf{A}^k)^{-1} \mathbf{g}\mathbf{g}^T (\mathbf{A}^k)^{-1}.$$

Technical details of the derivations can be found in the appendix. It is important to note here that Eq. (11.11) is obtained by solving the optimization problem (11.9) without considering the positivity constraint on $\mathbf{A}$. Despite this, (when $\mu\eta < \frac{1}{2}$) the obtained $\mathbf{A}$ is symmetric and positive, and hence the learned matrix-valued multi-view kernel is valid.

The positivity in block-sparse situation is not quite as straightforward. We note that even if we begin iteration with positive definite (pd) matrix the next iteration is not guaranteed to be always pd, and this is the reason for omitting the positivity constraint in the formulation of sparse problem. Nevertheless, all block-diagonal results are pd, and so are other results if certain conditions hold. In the experiments, we have observed that the solution is positive semidefinite. The full derivation of the proximal algorithm and notes about positiveness of $\mathbf{A}$ are in section "MVML Optimization" in the Appendix.

11.3.4 Regression: Squared Loss

Let us consider regression framework and apply squared loss function to (11.9). Let us denote $\mathbf{y}$ the vector of labels y_i. We get

$$\min_{\mathbf{A},\mathbf{g}} \quad \left\| \mathbf{y} - \left(\mathbf{w}^T \otimes \mathbf{I}_n\right)\mathbf{H}\mathbf{g} \right\|^2 + \lambda \left\langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \right\rangle + \eta \Omega(\mathbf{A}) \tag{11.14}$$

We already have solution for $\mathbf{A}$ with two regularizers $\Omega(\mathbf{A})$, thus we only consider in this section the problem:

$$\min_{\mathbf{g}} \quad \left\| \mathbf{y} - \left(\mathbf{w}^T \otimes \mathbf{I}_n\right)\mathbf{H}\mathbf{g} \right\|^2 + \lambda \left\langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \right\rangle. \tag{11.15}$$

Via a simple derivative, we arrive to the following solution for $\mathbf{g}$ with fixed $\mathbf{A}$:

$$\mathbf{g} = \left(\mathbf{H}\left(\mathbf{w}^T \otimes \mathbf{I}_n\right)^T \left(\mathbf{w}^T \otimes \mathbf{I}_n\right)\mathbf{H} + \lambda \mathbf{A}^\dagger\right)^{-1} \mathbf{H}\left(\mathbf{w}^T \otimes \mathbf{I}_n\right)^T \mathbf{y}. \tag{11.16}$$

If so desired, it is also possible to learn the weights $\mathbf{w}$. For fixed $\mathbf{g}$ and $\mathbf{A}$, the solution for $\mathbf{w}$ is

$$\mathbf{w} = \left(\mathbf{Z}^T \mathbf{Z}\right)^{-1} \mathbf{Z}^T \mathbf{y}, \tag{11.17}$$

where $\mathbf{Z} \in \mathbb{R}^{n \times v}$ is filled columnwise from $\mathbf{Hg}$.

The complexity of the algorithm is $\mathcal{O}(v^3n^3)$ for it computes the inverse of the $nv \times nv$ matrix $\mathbf{A}$, required for calculating $\mathbf{g}$. We will show later how to reduce the computational complexity of our algorithm via Nyström approximation, while conserving the desirable information about the multi-view problem.

11.3.5 Classification: Hinge Loss

Moving on to classification framework, we now require our labels to be in $\mathscr{Y} = \{-1, 1\}$.

Applying the SVM hinge loss function to (11.9), we get

$$\min_{\mathbf{A},\mathbf{g}} \sum_{i=1}^{n} \max\left(0, 1 - y_i \mathbf{w}^T \mathbf{H}_{x_i} \mathbf{g}\right) + \lambda \langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \rangle + \eta \Omega(\mathbf{A}). \tag{11.18}$$

To solve $\mathbf{g}$, we firstly introduce the slack variables ξ_i to the optimization problem, giving us the primal problem:

$$\begin{aligned} &\min_{\mathbf{g},\xi_i} \ \frac{1}{n} \sum_{i=1}^{n} \xi_i + \lambda \langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \rangle \\ &s.t. \ \ y_i \mathbf{w}^T \mathbf{H}_{x_i} \mathbf{g} \geq 1 - \xi_i \ \text{ and } \ \xi_i \geq 0 \quad \forall i. \end{aligned} \tag{11.19}$$

Solving this requires then writing the dual problem with Lagrangian multipliers, and then solving for the multipliers α_i from the quadratic problem:

$$\max_{\alpha_i} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2\lambda} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{w}^T \mathbf{H}_{\mathbf{x}_i} \mathbf{A} \mathbf{H}_{\mathbf{x}_j}^T \mathbf{w}, \qquad s.t. \ \ 0 \leq \alpha_i \leq \frac{1}{n}, \tag{11.20}$$

Having the α_i, we can calculate

$$\mathbf{g} = \frac{1}{2\lambda} \mathbf{A} \sum_{i=1}^{n} \alpha_i y_i \mathbf{H}_{\mathbf{x}_i}^T \mathbf{w}. \tag{11.21}$$

11.3.6 Illustration

We illustrate with simple toy data the effects of learning both within- and between-view metrics. We compare our method, MVML, to MKL that considers only within-view dependencies, and to output kernel learning (OKL) [9, 10] where separable kernels are learnt. We generated an extremely simple dataset of two classes and

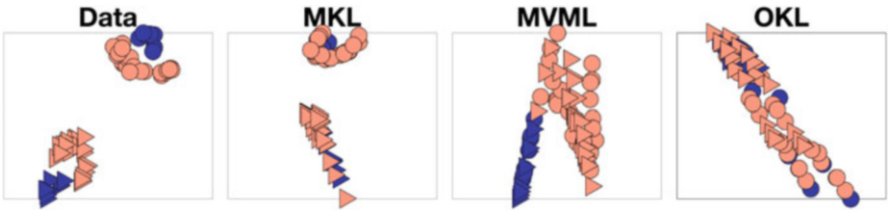


Fig. 11.1 Simple two-view dataset and its transformations—left: original data where one of the views is completely generated from the other by a linear transformation (a shear mapping followed by a rotation), left middle: MKL transformation, right middle: MVML transformation, and right: OKL transformation. MVML shows a linear separation of classes (blue/pale red) of the views (circles/triangles), while MKL and OKL do not

two views in $\mathbb{R}^2$, allowing for visualization and understanding of the way the methods perform classification with multi-view data. The second view in the dataset is completely generated from the first, through a linear transformation (a shear mapping followed by a rotation). The generated data and transformation arising from applying the algorithms are shown in Fig. 11.1. The space for transformed data is $\mathbb{R}^2$ since we used linear kernels for simplicity. Our MVML is the only method giving linear separation of the two classes. This means that it groups the data points into groups based on their class, not view, and thus is able to construct a good approximation of the initial data transformations by which we generated the second view.

11.3.7 Rademacher Complexity Bound

We now provide a generalization analysis of MVML algorithm using Rademacher complexities [2]. The notion of Rademacher complexity has been generalizable to vector-valued hypothesis spaces [22, 29, 33]. Previous work has analyzed the case where the matrix-valued kernel is fixed prior to learning, while our analysis considers the kernel learning problem. It provides a Rademacher bound for our algorithm when both the vector-valued function f and the metric between views $\mathbf{A}$ are learnt. We start by recalling that the feature map associated to the matrix-valued kernel K is the mapping $\Gamma : \mathscr{X} \to \mathscr{L}(\mathscr{Y}, \mathscr{H})$, where $\mathscr{X}$ is the input space, $\mathscr{Y} = \mathbb{R}^v$, and $\mathscr{L}(\mathscr{Y}, \mathscr{H})$ is the set of bounded linear operators from $\mathscr{Y}$ to $\mathscr{H}$ (see, e.g., [8, 23] for more details). It is known that $K(x, z) = \Gamma(x)^* \Gamma(z)$. We denote by $\Gamma_{\mathbf{A}}$ the feature map associated to our multi-view kernel (Eq. (11.5)). The hypothesis class of MVML is

$$\mathscr{H}_\lambda = \{x \mapsto f_{u,\mathbf{A}}(x) = \Gamma_{\mathbf{A}}(x)^* u : \mathbf{A} \in \Delta, \|u\|_{\mathscr{H}} \leq \beta\},$$

with $\Delta = \{\mathbf{A} : \mathbf{A} \succ 0, \|\mathbf{A}\|_F \leq \alpha\}$ and β is a regularization parameter. Let $\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_v$ be an iid family of vectors of independent Rademacher variables where $\boldsymbol{\sigma}_i \in \mathbb{R}^v$, $\forall i = 1, \ldots, n$. The empirical Rademacher complexity of the vector-valued class $\mathscr{H}_\lambda$ is the function $\hat{\mathscr{R}}_n(\mathscr{H}_\lambda)$ defined as:

$$\hat{\mathscr{R}}_n(\mathscr{H}_\lambda) = \frac{1}{n}\mathbb{E}\left[\sup_{f \in \mathscr{H}} \sup_{\mathbf{A} \in \Delta} \sum_{i=1}^{n} \boldsymbol{\sigma}_i^\top f_{u,\mathbf{A}}(x_i)\right].$$

Theorem 3 *The empirical Rademacher complexity of $\mathscr{H}_\lambda$ can be upper bounded as follows:*

$$\hat{\mathscr{R}}_n(\mathscr{H}_\lambda) \leq \frac{\beta\sqrt{\alpha\|q\|_1}}{n},$$

where $q = \left(tr(\mathbf{K}_l^2)\right)_{l=1}^v$, and $\mathbf{K}_l$ is the Gram matrix computed from the training set $\{x_1, \ldots, x_n\}$ with the kernel k_l defined on the view l. For kernels k_l such that $tr(\mathbf{K}_l^2) \leq \tau n$, we have

$$\hat{\mathscr{R}}_n(\mathscr{H}_\lambda) \leq \beta\sqrt{\frac{\alpha\tau v}{n}}.$$

The proof for the theorem can be found in the section "Proof of Theorem 3" in the Appendix. Using well-known results [26, chapters 8, 10], this bound on Rademacher complexity can be used to obtain generalization bounds for our algorithm. It is worth mentioning that in our multi-view setting the matrix-valued kernel is computed from the product of the kernel matrices defined over the views. This is why, our assumption is on the trace of the square of the kernel matrices $\mathbf{K}_l$. It is more restrictive than the usual one in the one-view setting ($tr(\mathbf{K}_l) \leq \tau n$), but is satisfied in some cases, like, for example, for diagonally dominant kernel matrices [30]. We note that although our assumptions are more restrictive than usual, they are not artificial, and even give further validation to our work, as we are in one sense using empirical feature maps in our kernel, which are described as solution when kernel matrices are diagonally dominant [30]. However, it might be interesting in future work to investigate whether it would be possible to obtain this bound with less restrictive constraints.

We note that the hypothesis class considered in this Rademacher bound requires that $\|\mathbf{A}\|_F \leq \alpha$. This is taken straight from the first version of our regularizer (11.10), but this holds also for our second regularizer.

11.3.8 Computational Efficiency via Nyström Approximation

As a way to reduce the complexity of the required computations, we use Nyström approximation on each one-view kernel matrix. In Nyström approximation method [35], a (scalar-valued) kernel matrix $\mathbf{M}$ is divided into four blocks:

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \\ \mathbf{M}_{21} & \mathbf{M}_{22} \end{bmatrix},$$

and is approximated by $\mathbf{M} \approx \mathbf{Q}\mathbf{W}^{\dagger}\mathbf{Q}^{\top}$, where $\mathbf{Q} = \begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} \end{bmatrix}^{\top}$ and $\mathbf{W} = \mathbf{M}_{11}$. Denote p as the number of rows of $\mathbf{M}$ chosen to build $\mathbf{W}$. This scheme gives a low-rank approximation of $\mathbf{M}$ by sampling p examples, and only the last block, $\mathbf{M}_{22}$, will be approximated.

We could approximate the block kernel matrix $\mathbf{G}$ directly by applying the Nyström approximation, but this would have the effect of removing the block structure in the kernel matrix and consequently the useful multi-view information might be lost. Instead, we proceed in a way that is consistent with the multi-view problem and approximate each kernel matrix defined over one view as $\mathbf{K}_l \approx \mathbf{Q}_l\mathbf{W}_l^{\dagger}\mathbf{Q}_l^{\top} = \mathbf{Q}_l(\mathbf{W}_l^{\dagger})^{1/2}(\mathbf{W}_l^{\dagger})^{1/2}\mathbf{Q}_l^{\top} = \mathbf{U}_l\mathbf{U}_l^{\top}, \forall\, l = 1, \ldots, v$. The goodness of approximation is based on the p chosen. Before performing the approximation, a random ordering of the samples is calculated. We note that in our multi-view setting we have to impose the same ordering over all the views.

We introduce the Nyström approximation to all our single-view kernels and define $\mathbf{U} = \text{blockdiag}(\mathbf{U}_1, \ldots, \mathbf{U}_v)$. We can now approximate our multi-view kernel (11.6) as:

$$\mathbf{G} = \mathbf{H}\mathbf{A}\mathbf{H} \approx \mathbf{U}\mathbf{U}^{\top}\mathbf{A}\mathbf{U}\mathbf{U}^{\top} = \mathbf{U}\tilde{\mathbf{A}}\mathbf{U}^{\top},$$

where we have written $\tilde{\mathbf{A}} = \mathbf{U}^{\top}\mathbf{A}\mathbf{U}$. Using this scheme, we obtain a block-wise Nyström approximation of $\mathbf{G}$ that preserves the multi-view structure of the kernel matrix while allowing substantial computational gains.

Let us write the optimization problem with approximation:

$$\min_{\tilde{\mathbf{A}}, \tilde{\mathbf{g}}} \sum_{i=1}^{n} V\left(y_i, \mathbf{w}^{\top}\mathbf{U}_{x_i}\tilde{\mathbf{g}}\right) + \lambda\langle\tilde{\mathbf{g}}, \tilde{\mathbf{A}}^{\dagger}\tilde{\mathbf{g}}\rangle + \eta\Omega(\tilde{\mathbf{A}}). \tag{11.22}$$

We note that the optimization problem is not strictly equivalent to the one before; namely, we impose the regularization over $\tilde{\mathbf{A}}$ and solve for it rather than for $\mathbf{A}$.

The problems will be solved as before, now just using $\mathbf{U}$ instead of $\mathbf{H}$ and iterating over $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{g}}$. The complexity is now of order $\mathcal{O}(v^3p^3)$ rather than $\mathcal{O}(v^3n^3)$, where $p \ll n$ is the number of samples chosen for the Nyström approximation in each block. From the obtained solution, it is possible to calculate the original $\mathbf{g}$ and $\mathbf{A}$ if needed.

To also reduce the complexity of predicting with our multi-view kernel framework, our block-wise Nyström approximation is used again on the test kernel matrices $\mathbf{K}_s^{test}$ computed with the test examples. With this scheme, it is not necessary for us to calculate the original $\mathbf{g}$ or $\mathbf{A}$. Let us recall that for each of our single-view kernels, we have an approximation $\mathbf{K}_s \approx \mathbf{U}_s\mathbf{U}_s^\top = \mathbf{Q}_s\mathbf{W}_s^\dagger\mathbf{Q}_s^\top$. We choose $\mathbf{Q}_s^{test}$ to be p first columns of the matrix $\mathbf{K}_s^{test}$, and define the approximation for the test kernel to be

$$\mathbf{K}_s^{test} \approx \mathbf{Q}_s^{test}\mathbf{W}_s^\dagger\mathbf{Q}_s^\top = \mathbf{Q}_s^{test}\left(\mathbf{W}_s^\dagger\right)^{1/2}\mathbf{U}_s^T.$$

In such an approximation, the error is in the last $n - p$ columns of $\mathbf{K}_s^{test}$. We gain in complexity, since if we were forced to use the test kernel as is, we would need to calculate $\mathbf{A}$ from $\tilde{\mathbf{A}}$ in $\mathcal{O}(vn^3)$ operations.

Solving $\tilde{\mathbf{A}}$ We obtain solutions for $\tilde{\mathbf{A}}$ from (11.22) exactly the same way as with full kernel matrices. The obtained solution with (11.10) will again satisfy the positivity condition when $\mu\eta < \frac{1}{2}$. For the sparse solution, the positivity is not always guaranteed but is achieved if certain conditions hold (see the appendix).

The update rule for $\tilde{\mathbf{A}}$ with (11.10) is now

$$\tilde{\mathbf{A}}^{k+1} = (1 - 2\mu\eta)\,\tilde{\mathbf{A}}^k + \mu\lambda(\tilde{\mathbf{A}}^k)^\dagger\tilde{\mathbf{g}}\tilde{\mathbf{g}}^\top(\tilde{\mathbf{A}}^k)^\dagger, \tag{11.23}$$

as for $\tilde{\mathbf{A}}$ with (11.12) we use

$$[\tilde{\mathbf{A}}^{k+1}]_\gamma = \left(1 - \frac{\eta}{\left\|[\tilde{\mathbf{A}}^k - \mu^k\nabla h(\tilde{\mathbf{A}}^k)]_\gamma\right\|_F}\right)_+ \left[\tilde{\mathbf{A}}^k - \mu^k\nabla h(\tilde{\mathbf{A}}^k)\right]_\gamma, \tag{11.24}$$

where μ^k is the step size,

$$h(\tilde{\mathbf{A}}^k) = \lambda\left\langle\tilde{\mathbf{g}}, (\mathbf{A}^k)^\dagger\tilde{\mathbf{g}}\right\rangle \qquad \text{and}$$
$$\nabla h(\tilde{\mathbf{A}}^k) = -\lambda(\tilde{\mathbf{A}}^k)^{-1}\tilde{\mathbf{g}}\tilde{\mathbf{g}}^\top(\tilde{\mathbf{A}}^k)^{-1}.$$

Solving Regression Problem The solution for regression framework as introduced in Sect. 11.3.4 updated to use Nyström approximated kernels is

$$\tilde{\mathbf{g}} = \left(\mathbf{U}^\top\left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)^\top\left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)\mathbf{U} + \lambda\tilde{\mathbf{A}}^\dagger\right)^{-1}\mathbf{U}^\top\left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)^\top\mathbf{y}. \tag{11.25}$$

Note that inverse required in solution is now over matrix of size $pv \times pv$ rather than $nv \times nv$ where $p \ll n$.

We can again calculate closed-form solution for $\mathbf{w}$, too, yielding

$$\mathbf{w} = (\tilde{\mathbf{Z}}^\top\tilde{\mathbf{Z}})^{-1}\tilde{\mathbf{Z}}^\top\mathbf{y}, \tag{11.26}$$

where $\tilde{\mathbf{Z}} \in \mathbb{R}^{n\times v}$ is filled columnwise from $\mathbf{U}\tilde{\mathbf{g}}$.

Solving Classification Problem As before, applying the Nyström approximated matrices is straightforward, and gives us the following optimization problem for variables α_i:

$$\max_{\alpha_i} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2\lambda} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{w}^T \mathbf{U}_{\mathbf{x}_i} \tilde{\mathbf{A}} \mathbf{U}_{\mathbf{x}_j}^T \mathbf{w}, \qquad s.t. \;\; 0 \leq \alpha_i \leq \frac{1}{n}, \tag{11.27}$$

and the solution for $\tilde{\mathbf{g}}$

$$\tilde{\mathbf{g}} = \frac{1}{2\lambda} \tilde{\mathbf{A}} \sum_{i=1}^{n} \alpha_i y_i \mathbf{U}_{\mathbf{x}_i}^T \mathbf{w}. \tag{11.28}$$

Note that with this formulation the benefit of Nyström approximation is not easy to see as we do not explicitly have to calculate the inverse of $\tilde{\mathbf{A}}$, where the gains for complexity would be most substantial (inverse of $mv \times mv$ matrix instead of $np \times np$ matrix). However by choosing the optimization procedure for the quadratic problem carefully, it can be exploited to get improvements in computational cost, mainly coming from slightly smaller matrix multiplications. Note that if extended to semi-supervised case (next section), we will again need to calculate the inverse of $\tilde{\mathbf{A}}$, and the computational gains are more clear.

11.3.9 Extension to Semi-supervised Case

Although the method presented in this chapter is for supervised learning context, it is possible to extend into semi-supervised setting by adding a manifold regularization term.

We consider the term introduced in [25]:

$$v_i \langle \mathbf{f}, \mathbf{M}\mathbf{f} \rangle = v_B \langle \mathbf{f}, \mathbf{M}_B \mathbf{f} \rangle + v_W \langle \mathbf{f}, \mathbf{M}_W \mathbf{f} \rangle \tag{11.29}$$

Here, $\mathbf{f} = (f(x_1)^\top, \ldots, f(x_{n+u})^\top)^\top$ and $\mathbf{M}$ (and matrices $\mathbf{M}_B$ and $\mathbf{M}_W$) is a symmetric, positive operator, in this case a matrix of size $(n+u)v \times (n+u)v$ where u is the number of unlabeled samples available.

The first term is for between-view regularization, and it measures the consistency of the component functions across different views. The second term is for within-view regularization measuring the smoothness of the component functions in their corresponding views. In [25], some concrete realizations of the matrices M_B and M_W are introduced.

Let us recall that

$$f(z) = \sum_{i=1}^{n} K(z, x_i)c_i = \mathbf{H}_z^{n+u}\mathbf{A}\mathbf{H}\mathbf{c} = \mathbf{H}_z^{n+u}\mathbf{g}$$

where $\mathbf{H}^{n+u}$ is a matrix of size $(n+u)v \times nv$. It is the kernel matrix where all labeled and unlabeled samples will be evaluated in the kernel against the labeled samples.

Note that $\mathbf{f}$ is arranged differently from our other vectors and matrices. It follows ordering by samples; that is, it is made up of n vectors of length v when our other matrices and vectors follow ordering by views and contain parts of size $n \times n$ (or length n). It is thus the case that we will have to rearrange matrix $\mathbf{M}$ to suit our framework. We denote this as $\tilde{\mathbf{M}}$. When we do this, the regularization term is

$$\nu_i \langle \mathbf{H}^{n+u}\mathbf{g}, \tilde{\mathbf{M}}\mathbf{H}^{n+u}\mathbf{g} \rangle = \nu_B \langle \mathbf{H}^{n+u}\mathbf{g}, \tilde{\mathbf{M}}_B\mathbf{H}^{n+u}\mathbf{g} \rangle + \nu_W \langle \mathbf{H}^{n+u}\mathbf{g}, \tilde{\mathbf{M}}_W\mathbf{H}^{n+u}\mathbf{g} \rangle \tag{11.30}$$

Adding this (or these) regularizer to our problem with the change of variables does not change the solutions for metric matrix $\mathbf{A}$, but affect only $\mathbf{g}$. We get in this case the solution for regression framework:

$$\mathbf{g} = \big(\mathbf{H}\big(\mathbf{w}^\top \otimes \mathbf{I}_n\big)^\top \big(\mathbf{w}^\top \otimes \mathbf{I}_n\big)\mathbf{H} + \lambda\mathbf{A}^\dagger + \nu\big(\mathbf{H}^{n+u}\big)^\top \tilde{\mathbf{M}}\mathbf{H}^{n+u}\big)^{-1}\mathbf{H}\big(\mathbf{w}^\top \otimes \mathbf{I}_n\big)^\top \mathbf{y}. \tag{11.31}$$

and the approximated version similarly.

For SVM framework, the new optimization problem with respect to the dual variables α_i is

$$\max_{\alpha_i} \quad \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j y_i y_j \mathbf{w}^T \mathbf{H}_{x_i}[\lambda\mathbf{A}^\dagger + \nu(\mathbf{H}^{n+u})^\top \tilde{\mathbf{M}}\mathbf{H}^{n+u}]^{-1}\mathbf{H}_{x_j}^T\mathbf{w} \tag{11.32}$$

$$s.t.\ \ 0 \le \alpha_i \le \frac{1}{l}.$$

and the solution given the parameters α_i is now

$$\mathbf{g} = \frac{1}{2}\big[\lambda\mathbf{A}^\dagger + \nu(\mathbf{H}^{n+u})^\top \tilde{\mathbf{M}}\mathbf{H}^{n+u}\big]^{-1}\sum_{i=1}^{l}\alpha_i y_i \mathbf{H}_{x_i}^T\mathbf{w}. \tag{11.33}$$

It is possible to apply Nyström approximation also with manifold regularization, either the same way as before or it can be calculated straight away with the whole data and extract appropriate parts when needed. Applied to whole data, the approximation on training data also in a sense might depend on test data, but as we are considering manifold regularization we are in any case using the whole dataset when learning.

11.4 Experiments

Here, we evaluate the proposed multi-view metric learning (MVML) method on real-world datasets and compare it to relevant methods. The chosen datasets are "pure" multi-view datasets; that is to say, the view division arises naturally from the data.

We perform two sets of experiments with two goals. First, we evaluate our method in regression setting with a large range of Nyström approximation levels in order to understand the effect it has on our algorithm. Secondly, we compare MVML to relevant state-of-the-art methods in classification. In both cases, we use non-multi-view methods to justify the multi-view approach. The methods we use in addition to our own[2] (**MVML** and **MVMLsparse**) are:

- **MVML_Cov and MVML_I**: we use pre-set kernels in our framework: MVML_Cov uses the kernel from [16] and MVML_I refers to the case when we have only one-view kernel matrices in the diagonal of the multi-view kernel.
- **lpMKL** is an algorithm for learning weights for MKL kernel [18]. We apply it to kernel regression.
- **OKL** [9, 10] is a kernel learning method for separable kernels.
- **MLKR** [34] is an algorithm for metric learning in kernel setting.
- **MUMBO** [19] is a multi-class boosting-based multi-view algorithm which is intended to reinforce the cooperation among views.
- **KRR and SVM**: We use kernel ridge regression and support vector machines with one-view as well as in early fusion (ef) and late fusion (lf) in order to validate the benefits of using multi-view methods.

We perform our experiments with Python, but for OKL and MLKR we use the MATLAB codes provided by authors.[3] In MVML, we set weights uniformly to $\frac{1}{v}$. For all the datasets, we use Gaussian kernels, $k(\mathbf{x}, \mathbf{z}) = exp(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{z}\|^2)$.

11.4.1 Effect of Nyström Approximation

For our first experiment, we consider SARCOS-dataset,[4] where the task is to map a 21-dimensional input space (7 joint positions, 7 joint velocities, and 7 joint accelerations) to the corresponding 7 joint torques. Here, we present results to the first task.

The results with various levels of Nyström approximation—averaged over four approximations—from 1% to 100% of data are shown in Fig. 11.2. Regulariza-

[2] Code for MVML is available at https://lives.lif.univ-mrs.fr/?page_id=12.

[3] https://www.cs.cornell.edu/~kilian/code/code.html and https://github.com/cciliber/matMTL.

[4] http://www.gaussianprocess.org/gpml/data.

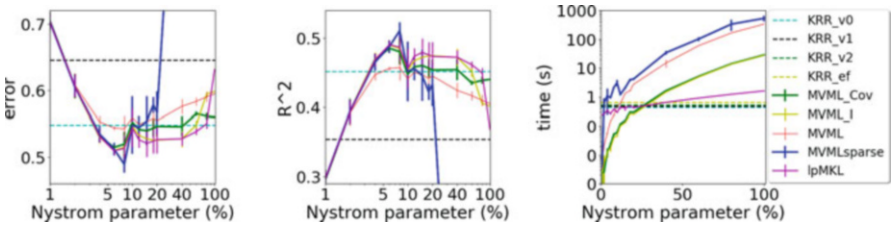


Fig. 11.2 Regression on Sarcos1-dataset. Left: normalized mean-squared errors (the lower the better), middle: R^2-score (the higher the better), right: running times; as functions of Nyström approximation level (note the logarithmic scales). The results for KRR are calculated without approximation and are shown as horizontal dashed lines. Results for view 2 and early fusion are worse than others and outside of the scope of the two plots

tion parameters were cross-validated over values $\lambda \in [1e{-}08, 10]$ and $\eta \in [1e{-}04, 100]$. Kernel parameter $\gamma = 1/2\sigma^2$ was fixed to be 1/number of features as a trade-off between overfitting and underfitting. We used only 1000 data samples of the available 44,484 in training (all 4449 in testing) to be feasibly able to show the effect of approximating the matrices on all levels, and wish to note that using more data samples with moderate approximation level we can yield a lower error than presented here: for example, with 2000 training samples and Nyström approximation level of 8%, we obtain an error of 0.3915. However, the main goal of our experiment was to see how our algorithm behaves with various Nyström approximation levels and because of the high complexity of our algorithm trained on the full dataset without approximation we performed this experiment with low amount of samples.

The lowest error was obtained with our MVMLsparse algorithm at 8% Nyström approximation level. All the multi-view results seem to benefit from using the approximation. Indeed, approximating the kernel matrices can be seen as a form of regularization and our results reflect on that [28]. Overall, our MVML learning methods have much higher computational cost with large Nyström parameters, as can be seen from Fig. 11.2, rightmost plot. However with smaller approximation levels with which the methods are intended to be used, the computing time is competitive.

11.4.2 Classification Results

In our classification experiments, we use three real-world multi-view datasets:

– Flower17[5] (7 views, 17 classes, and 80 samples per class)

[5] http://www.robots.ox.ac.uk/~vgg/data/flowers/17.

Table 11.1 Classification accuracies ± standard deviation

Method	MVML	MVMLsp	MVMLCov	MVML_I	OKL	MLKR
Flower17 (6%)	75.98 ± 2.62	75.71 ± 2.48	75.71 ± 2.19	**76.96 ± 2.35**	68.73 ± 1.95	63.82 ± 2.51
Flower17 (12%)	77.89 ± 2.41	77.43 ± 2.44	77.30 ± 2.36	**78.36 ± 2.52**	75.19 ± 1.97	64.41 ± 2.41
Flower17 (24%)	78.60 ± 1.41	78.60 ± 1.36	79.00 ± 1.75	**79.19 ± 1.51**	76.76 ± 1.62	65.44 ± 1.36
uWaveG. (6%)	93.13 ± 0.18	**93.29 ± 0.15**	92.37 ± 0.22	91.63 ± 0.33	70.09 ± 1.07	71.09 ± 0.94
uWaveG. (12%)	**93.03 ± 0.11**	92.86 ± 0.26	92.53 ± 0.18	92.59 ± 0.13	74.07 ± 0.26	80.22 ± 0.38
uWaveG. (24%)	**93.32 ± 0.04**	93.26 ± 0.15	92.66 ± 0.05	93.10 ± 0.11	76.65 ± 0.33	86.38 ± 0.31
Leaves (6%)	83.57 ± 3.38	**83.59 ± 3.38**	83.57 ± 3.38	83.55 ± 3.41	61.62 ± 1.58	72.98 ± 3.00
Leaves (12%)	83.81 ± 3.78	84.91 ± 3.51	84.75 ± 2.89	**85.26 ± 3.02**	68.27 ± 1.62	73.99 ± 2.01
Leaves (24%)	81.74 ± 3.78	85.61 ± 3.77	84.97 ± 3.10	**85.73 ± 3.42**	71.80 ± 2.04	74.07 ± 2.41

Method	lpMKL	MUMBO	efSVM	lfSVM	1 view SVM
Flower17 (6%)	75.54 ± 2.61	75.27 ± 2.32	–	15.32 ± 1.94	11.59 ± 1.54
Flower17 (12%)	77.87 ± 2.52	76.25 ± 2.11	–	23.82 ± 2.38	15.74 ± 1.54
Flower17 (24%)	78.75 ± 1.58	76.47 ± 1.42	–	38.24 ± 2.31	22.79 ± 0.79
uWaveG. (6%)	92.34 ± 0.18	90.08 ± 0.42	80.00 ± 0.74	71.24 ± 0.41	56.54 ± 0.38
uWaveG. (12%)	92.48 ± 0.21	90.45 ± 0.47	82.29 ± 0.63	72.53 ± 0.16	57.50 ± 0.17
uWaveG. (24%)	92.85 ± 0.13	90.68 ± 0.33	84.07 ± 0.23	72.99 ± 0.06	58.01 ± 0.05
Leaves (6%)	83.57 ± 3.27	79.10 ± 2.29	55.19 ± 2.85	41.22 ± 2.31	51.26 ± 2.58
Leaves (12%)	85.10 ± 3.29	78.58 ± 2.44	59.38 ± 3.27	42.85 ± 2.21	53.41 ± 2.16
Leaves (24%)	85.68 ± 3.21	77.90 ± 2.68	62.09 ± 3.32	43.69 ± 2.27	54.43 ± 2.39

For MVML, we provide the best result given by the two loss functions. The number after the dataset indicates the level of Nyström approximation on the kernels. The results for efSVM classification for Flower17-dataset are missing as only similarity matrices for each view were provided. Last column reports the best result obtained when using only one view

– uWaveGesture[6] (3 views, 8 classes, 896 data samples for training, and 3582 samples for testing)
– 100 Leaves[7] (3 views, 100 classes, and 16 samples per class)

For the 100 Leaves dataset, we reduced the number of classes to 34 by considering only the genus, so, for example, both "Alnus Rubra" and "Alnus Sieboldiana" are members of the same class, "Alnus." As there was one missing value in one of the views for "Acer Campestre," we do not consider those samples at all, so in total the data contains 1584 samples. For all the experiments, we used Gaussian kernels and set the kernel parameter to be mean of distances, $\sigma = \frac{1}{n^2}\sum_{i,j=1}^{n} \|\mathbf{x}_i - \mathbf{x}_j\|$. The regularization parameters for Flower17 and uWaveGesture were obtained by cross-validation over values $\lambda \in [1e{-}08, 10]$ and $\eta \in [1e{-}03, 100]$ over training data. The results are averaged over four approximations.

We adopted one-vs-all classification approach for multiclass classification (except for MUMBO). The results are displayed in Table 11.1. The MVML results are always notably better than the SVM results, or the results obtained with OKL or MLKR. Compared to MVML, OKL and MLKR accuracies decrease more with low approximation levels. We can see that all MVML methods perform very similarly, sometimes the best result is obtained with fixed multi-view kernel, sometimes when $\mathbf{A}$ is learned.

As an example of our sparse output with MVML, we note that running the algorithm with Flower17 dataset with 12% approximation often resulted in an spd matrix as in Fig. 11.3. Indeed, the resulting sparsity is very interesting and tells us about the importance of the views and their interactions.

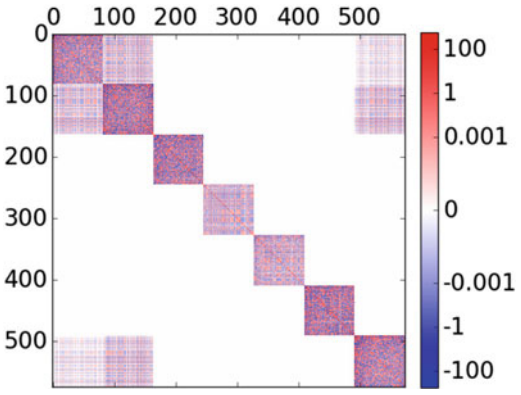


Fig. 11.3 An example of learned $\tilde{\mathbf{A}}$ with MVMLsparse from Flower17 (12%) experiments

[6] http://www.cs.ucr.edu/~eamonn/time_series_data.
[7] https://archive.ics.uci.edu/ml/datasets/One-hundred+plant+species+leaves+data+set.

11.5 Conclusion

This chapter has introduced a general class of matrix-valued multi-view kernels for which we have presented multiple ways for simultaneously learning a multi-view function and a metric in vector-valued kernel spaces. We provided an iterative algorithm for the resulting optimization problems (two loss functions, and two regularizers for metric matrix), and have been able to significantly lower the high computational cost associated with kernel methods by introducing block-wise Nyström approximation. We have also given a possible extension for applying our method to semi-supervised setting.

We have explained the feasibility of our approach onto a trivial dataset which reflects the objective of learning the within-view and between-view correlation metrics. The performance of our approach was illustrated with experiments with real multi-view datasets by comparing our method to standard multi-view approaches, as well as methods for metric learning and kernel learning. Our sparse method is especially appealing in the sense that it gives us very intuitive information about the importance of the views.

Acknowledgements We thank the anonymous reviewers for their relevant and helpful comments. This work is granted by French ANR project Lives (ANR-15-CE23-0026).

Appendix

MVML Optimization

Here, we go through the derivations of the solutions $\mathbf{A}$, $\mathbf{g}$, and $\mathbf{w}$ for our optimization problems. The presented derivations are for the case without Nyström approximation or manifold regularization term; however, the derivations with those follow the same idea.

Solving for A with (11.10)

When we consider $\mathbf{g}$ (and $\mathbf{w}$) to be fixed in the MVML framework (11.9), for $\mathbf{A}$ with regularizer (11.10) we have the following minimization problem:

$$\min_{\mathbf{A}} \quad \lambda \left\langle \mathbf{g}, \mathbf{A}^{\dagger}\mathbf{g} \right\rangle + \eta \|\mathbf{A}\|_F^2$$

Derivating this with respect to $\mathbf{A}$ gives us[8]

$$\begin{aligned}
&\frac{d}{d\mathbf{A}}\ \lambda\left\langle \mathbf{g}, \mathbf{A}^{\dagger}\mathbf{g}\right\rangle + \eta\|\mathbf{A}\|_F^2 \\
&\quad = \frac{d}{d\mathbf{A}}\ \lambda\left\langle \mathbf{g}, \mathbf{A}^{\dagger}\mathbf{g}\right\rangle + \eta\ tr(\mathbf{A}\mathbf{A}) \\
&\quad = -\lambda\mathbf{A}^{\dagger}\mathbf{g}\mathbf{g}^{\top}\mathbf{A}^{\dagger} + 2\eta\mathbf{A}
\end{aligned}$$

Thus, the gradient descent step will be

$$\mathbf{A}^{k+1} = (1 - 2\mu\eta)\,\mathbf{A}^{k} + \mu\lambda\left(\mathbf{A}^{k}\right)^{\dagger}\mathbf{g}\mathbf{g}^{\top}\left(\mathbf{A}^{k}\right)^{\dagger}$$

when moving to the direction of negative gradient with step size μ.

Solving for A with Sparse Regularizer (11.12)

To solve $\mathbf{A}$ with the block-sparse regularization, we use proximal minimization. Let us recall the optimization problem after the change of the variable:

$$\min_{\mathbf{A}}\ \ \lambda\langle \mathbf{g}, \mathbf{A}^{\dagger}\mathbf{g}\rangle + \eta\sum_{\gamma\in\mathscr{G}}\|\mathbf{A}_{\gamma}\|_F,$$

and denote

$$h(\mathbf{A}) = \lambda\left\langle \mathbf{g}, \mathbf{A}^{\dagger}\mathbf{g}\right\rangle$$

and

$$\Omega(\mathbf{A}) = \eta\sum_{\gamma\in\mathscr{G}}\|\mathbf{A}_{\gamma}\|_F$$

for the two terms in our optimization problem that contain the matrix $\mathbf{A}$.

Without going into the detailed theory of proximal operators and proximal minimization, we remark that the proximal minimization algorithm update takes the form:

$$\mathbf{A}^{k+1} = \mathbf{prox}_{\mu^k\Omega}(\mathbf{A}^{k} - \mu^k\nabla h(\mathbf{A}^{k})).$$

[8]Here we have used Equation 62 from Matrix cookbook (https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf).

It is well known that in traditional group-lasso situation the proximal operator is

$$[\mathbf{prox}_{\mu^k \Omega}(\mathbf{z})]_\gamma = \left(1 - \frac{\eta}{\|\mathbf{z}_\gamma\|_2}\right)_+ \mathbf{z}_\gamma,$$

where $\mathbf{z}$ is a vector and $+$ denotes the maximum of zero and the value inside the brackets. In our case, we are solving for a matrix, but due to the equivalence of Frobenius norm to vector 2-norm we can use this exact same operator. Thus, we get as the proximal update:

$$[\mathbf{A}^{k+1}]_\gamma = \left(1 - \frac{\eta}{\|[\mathbf{A}^k - \mu^k \nabla h(\mathbf{A}^k)]_\gamma\|_F}\right)_+ [\mathbf{A}^k - \mu^k \nabla h(\mathbf{A}^k)]_\gamma,$$

where

$$\nabla h(\mathbf{A}^k) = -\lambda (\mathbf{A}^k)^{-1} \mathbf{g}\mathbf{g}^\top (\mathbf{A}^k)^{-1}.$$

We can see from the update formula and the derivative that if $\mathbf{A}^k$ is a positive matrix, the update without block-multiplication, $\mathbf{A}^k - \mu^k \nabla h(\mathbf{A}^k)$, will be positive, too. This is unfortunately not enough to guarantee the general positivity of $\mathbf{A}^{k+1}$. However, we note that it is, indeed, positive if it is block-diagonal, and in general whenever a matrix of the multipliers α:

$$\alpha_{st} = \left(1 - \frac{\eta}{\|[\mathbf{A}^k - \mu^k \nabla h(\mathbf{A}^k)]_{st}\|_2}\right)_+$$

is positive, then $\mathbf{A}^{k+1}$ is, too (see [13] for reference—this is a block-wise Hadamard product where the blocks commute).

Solving for g and w in Regression Setting

Let us first focus on the case where $\mathbf{A}$ and $\mathbf{w}$ are fixed, and we solve for $\mathbf{g}$. We calculate the derivative of the expression in Eq. (11.14):

$$\begin{aligned}
&\frac{d}{d\mathbf{g}} \left\|\mathbf{y} - \left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)\mathbf{H}\mathbf{g}\right\|^2 + \lambda \left\langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \right\rangle \\
&\quad = \frac{d}{d\mathbf{g}} \left\langle \mathbf{y}, \mathbf{y} \right\rangle - 2\left\langle \mathbf{y}, \left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)\mathbf{H}\mathbf{g} \right\rangle + \left\langle \left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)\mathbf{H}\mathbf{g}, \left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)\mathbf{H}\mathbf{D} \right\rangle + \lambda \langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \rangle \\
&\quad = -2\mathbf{H}\left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)^\top \mathbf{y} + 2\mathbf{H}\left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)^\top \left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)\mathbf{H}\mathbf{g} + 2\lambda \mathbf{A}^\dagger \mathbf{g}
\end{aligned}$$

By setting this to zero, we obtain the solution:

$$\mathbf{g} = \left(\mathbf{H}\left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)^\top \left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)\mathbf{H} + \lambda \mathbf{A}^\dagger\right)^{-1} \mathbf{H}\left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)^\top \mathbf{y}.$$

As for $\mathbf{w}$ when $\mathbf{A}$ and $\mathbf{g}$ are fixed, we need only to consider optimizing:

$$\min_{\mathbf{w}} \quad \left\|\mathbf{y} - \left(\mathbf{w}^\top \otimes \mathbf{I}_n\right)\mathbf{H}\mathbf{g}\right\|^2.$$

If we denote that $\mathbf{Z} \in \mathbb{R}^{n \times v}$ is equal to reshaping $\mathbf{Hg}$ by taking the elements of the vector and arranging them onto the columns of $\mathbf{Z}$, we obtain the following form:

$$\min_{\mathbf{w}} \quad \|\mathbf{y} - \mathbf{Z}\mathbf{w}\|^2.$$

One can easily see by taking the derivative and setting it to zero that the solution for this is

$$\mathbf{w} = \left(\mathbf{Z}^\top \mathbf{Z}\right)^{-1} \mathbf{Z}^\top \mathbf{y}. \tag{11.17}$$

Solving the SVM Problem

Let us recall the optimization problem with hinge loss:

$$\min_{\mathbf{A},\mathbf{g}} \sum_{i=1}^{n} \max\left(0, 1 - y_i \mathbf{w}^T \mathbf{H}_{x_i} \mathbf{g}\right) + \lambda \langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \rangle + \eta \Omega(\mathbf{A}). \tag{11.18}$$

To solve $\mathbf{g}$, we firstly introduce the slack variables ξ_i to the optimization problem, giving us the primal problem:

$$\begin{aligned} \min_{\mathbf{g},\xi_i} \quad & \frac{1}{l} \sum_{i=1}^{l} \xi_i + \lambda \langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \rangle \\ s.t. \quad & y_i \mathbf{w}^T \mathbf{H}_{x_i} \mathbf{g} \geq 1 - \xi_i \quad \text{and} \quad \xi_i \geq 0 \quad \forall i. \end{aligned}$$

From this, the dual problem with Lagrangian multipliers is

$$L(\mathbf{g}, \xi, \alpha, \beta) = \frac{1}{l} \sum_{i=1}^{l} \xi_i + \lambda \langle \mathbf{g}, \mathbf{A}^\dagger \mathbf{g} \rangle - \sum_{i=1}^{l} \alpha_i \left[\xi_i - 1 + y_i \mathbf{w}^T \mathbf{H}_{x_i} \mathbf{g}\right] - \sum_{i=1}^{l} \beta_i \xi_i,$$

with constraint that the α_i and β_i have to be positive (dual feasibility).

From derivative with respect to ξ_i, we get

$$\frac{d}{d\xi_i} = \frac{1}{l} - \alpha_i - \beta_i = 0 \quad \Rightarrow \quad \alpha_i + \beta_i = \frac{1}{l}$$

and the derivative with respect to $\mathbf{g}$ gives us

$$\frac{d}{d\mathbf{g}} = 2\lambda \mathbf{A}^{\dagger}\mathbf{g} - \sum_{i=1}^{l} \alpha_i \mathbf{H}_{x_i}^T \mathbf{w} y_i = 0$$

$$\Rightarrow \quad \mathbf{g} = \frac{1}{2\lambda}\mathbf{A}\sum_{i=1}^{l} \alpha_i y_i \mathbf{H}_{x_i}^T \mathbf{w}. \tag{11.21}$$

The α_i are still unknown. When we substitute $\beta_i = 1/l - \alpha_i$ and the solution obtained for $\mathbf{g}$ while ignoring all the terms that do not depend of α_i, we are left with

$$\max_{\alpha_i} \quad \sum_{i=1}^{l} \alpha_i - \frac{1}{2\lambda} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j \mathbf{w}^T \mathbf{H}_{\mathbf{x}_i} \mathbf{A} \mathbf{H}_{\mathbf{x}_j}^T \mathbf{w}, \qquad s.t. \;\; 0 \leq \alpha_i \leq \frac{1}{n}, \tag{11.20}$$

which is a quadratic optimization problem and can be easily solved.

Proof of Theorem 3

Theorem 3 *Let $\mathscr{H}$ be a vector-valued RKHS associated with the multi-view kernel K defined by Eq. (11.5). Consider the hypothesis class $\mathscr{H}_\lambda = \{x \mapsto f_{u,\mathbf{A}}(x) = \Gamma_{\mathbf{A}}(x)^* u : \mathbf{A} \in \Delta, \|u\|_{\mathscr{H}} \leq \beta\}$, with $\Delta = \{\mathbf{A} : \mathbf{A} \succ 0, \|\mathbf{A}\|_F \leq \alpha\}$. The empirical Rademacher complexity of $\mathscr{H}_\lambda$ can be upper bounded as follows:*

$$\hat{\mathscr{R}}_n(\mathscr{H}_\lambda) \leq \frac{\beta\sqrt{\alpha \|q\|_1}}{n},$$

where $q = \left(tr(\mathbf{K}_l^2)\right)_{l=1}^{v}$, and $\mathbf{K}_l$ is the Gram matrix computed from the training set $\{x_1, \ldots, x_n\}$ with the kernel k_l defined on the view l. For kernels k_l such that $tr(\mathbf{K}_l^2) \leq \tau n$, we have

$$\hat{\mathscr{R}}_n(\mathscr{H}_\lambda) \leq \beta\sqrt{\frac{\alpha\tau v}{n}}.$$

Proof We start by recalling that the feature map associated to the operator-valued kernel K is the mapping $\Gamma : \mathscr{X} \to \mathscr{L}(\mathscr{Y}, \mathscr{H})$, where $\mathscr{X}$ is the input space, $\mathscr{Y} = \mathbb{R}^v$, and $\mathscr{L}(\mathscr{Y}, \mathscr{H})$ is the set of bounded linear operators from $\mathscr{Y}$ to $\mathscr{H}$ (see, e.g., [8, 23] for more details). It is known that $K(x, z) = \Gamma(x)^* \Gamma(z)$. We denote by $\Gamma_{\mathbf{A}}$ the feature map associated to our multi-view kernel (Eq. (11.5)). We also define the matrix $\boldsymbol{\Sigma} = (\boldsymbol{\sigma})_{i=1}^{n} \in \mathbb{R}^{nv}$

$$
\begin{aligned}
\hat{\mathscr{R}}_n(\mathscr{H}_\lambda) &= \frac{1}{n}\mathbb{E}\left[\sup_{f\in\mathscr{H}}\sup_{\mathbf{A}\in\Delta}\sum_{i=1}^{n}\boldsymbol{\sigma}_i^\top f_{u,\mathbf{A}}(x_i)\right] \\
&= \frac{1}{n}\mathbb{E}\left[\sup_{u}\sup_{\mathbf{A}}\sum_{i=1}^{n}\langle\boldsymbol{\sigma}_i, \Gamma_{\mathbf{A}}(x_i)^* u\rangle_{\mathbb{R}^v}\right] \\
&= \frac{1}{n}\mathbb{E}\left[\sup_{u}\sup_{\mathbf{A}}\sum_{i=1}^{n}\langle\Gamma_{\mathbf{A}}(x_i)\boldsymbol{\sigma}_i, u\rangle_{\mathscr{H}}\right] \quad (1) \\
&\leq \frac{\beta}{n}\mathbb{E}\left[\sup_{\mathbf{A}}\|\sum_{i=1}^{n}\Gamma_{\mathbf{A}}(x_i)\boldsymbol{\sigma}_i\|_{\mathscr{H}}\right] \quad (2) \\
&= \frac{\beta}{n}\mathbb{E}\left[\sup_{\mathbf{A}}\left(\sum_{i,j=1}^{n}\langle\boldsymbol{\sigma}_i, K_{\mathbf{A}}(x_i,x_j)\boldsymbol{\sigma}_j\rangle_{\mathbb{R}^v}\right)^{\frac{1}{2}}\right] \quad (3) \\
&= \frac{\beta}{n}\mathbb{E}\left[\sup_{\mathbf{A}}(\langle\boldsymbol{\Sigma}, \mathbf{G}_{\mathbf{A}}\boldsymbol{\Sigma}\rangle_{\mathbb{R}^{nv}})^{1/2}\right] \\
&= \frac{\beta}{n}\mathbb{E}\left[\sup_{\mathbf{A}}\langle\boldsymbol{\Sigma}, \mathbf{HAH}\boldsymbol{\Sigma}\rangle^{1/2}\right] \\
&= \frac{\beta}{n}\mathbb{E}\left[\sup_{\mathbf{A}} tr(\mathbf{H}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top\mathbf{HA})^{1/2}\right] \\
&\leq \frac{\beta}{n}\mathbb{E}\left[\sup_{\mathbf{A}} tr([\mathbf{H}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top\mathbf{H}]^2)^{1/4} tr(\mathbf{A}^2)^{1/4}\right] \quad (4) \\
&\leq \frac{\beta}{n}\mathbb{E}\left[\sup_{\mathbf{A}} tr(\mathbf{H}^2\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top)^{1/2} tr(\mathbf{A}^2)^{1/4}\right] \\
&\leq \frac{\beta\sqrt{\alpha}}{n}\mathbb{E}\left[\sup_{\mathbf{A}} tr(\mathbf{H}^2\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top)^{1/2}\right] \\
&= \frac{\beta\sqrt{\alpha}}{n}\mathbb{E}\left[tr(\mathbf{H}^2\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top)^{1/2}\right] \\
&\leq \frac{\beta\sqrt{\alpha}}{n}\left(\mathbb{E}\left[tr(\mathbf{H}^2\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top)\right]\right)^{1/2} \quad (5) \\
&= \frac{\beta\sqrt{\alpha}}{n}\left(tr\left[\mathbf{H}^2\mathbb{E}(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\top)\right]\right)^{1/2} \\
&= \frac{\beta\sqrt{\alpha}}{n}\sqrt{\|\left(tr\left(\mathbf{K_1}^2\right), \ldots, tr\left(\mathbf{K_v}^2\right)\right)\|_1}.
\end{aligned}
$$

Here, (1) and (3) are obtained with reproducing property, (2) and (4) with Cauchy-Schwarz inequality, and (5) with Jensen's inequality. The last equality follows from the fact that $tr(\mathbf{H}^2) = \sum_{l=1}^{v} tr(\mathbf{K_l}^2)$. For kernels k_l that satisfy $tr(\mathbf{K}_l^2) \leq \tau n$, $l = 1, \ldots, v$, we obtain that

$$\hat{\mathscr{R}}_n(\mathscr{H}_\lambda) \leq \beta \sqrt{\frac{\alpha \tau v}{n}}.$$

□

References

1. Alvarez, M.A., Rosasco, L., Lawrence, N.D., et al.: Kernels for Vectorvalued Functions: A Review. Foundations and Trends® in Machine Learning 4.3, pp. 195–266. Now Publishers, Delft (2012)
2. Bartlett, P.L., Mendelson, S.: Rademacher and Gaussian complexities: risk bounds and structural results. J. Mach. Learn. Res. **3**, 463–482 (2002)
3. Bellet, A., Habrard, A., Sebban, M.: Metric Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning 9.1, pp. 1–151. Morgan & Claypool Publishers, San Rafael (2015)
4. Blum, A.B., Mitchell, T.M.: Combining labeled and unlabeled data with co-training. In: 11th Annual Conference on Computational Learning Theory (COLT), pp. 92–100 (1998)
5. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2004)
6. Brouard, C., Szafranski, M., d'Alché Buc, F.: Input output kernel regression: supervised and semi-supervised structured output prediction with operator-valued kernels. J. Mach. Learn. Res. **17**(176), 1–48 (2016)
7. Caponnetto, A., Micchelli, C.A., Pontil, M., Ying, Y.: Universal multi-task kernels. J. Mach. Learn. Res. **9**, 1615–1646 (2008)
8. Carmeli, C., De Vito, E., Toigo, A., Umanita, V.: Vector valued reproducing kernel Hilbert spaces and universality. Anal. Appl. **08**(01), 19–61 (2010)
9. Ciliberto, C., Mroueh, Y., Poggio, T., Rosasco, L.: Convex learning of multiple tasks and their structure. In: International Conference in Machine Learning (ICML) (2015)
10. Dinuzzo, F., Ong, C.S., Pillonetto, G., Gehler, P.V.: Learning output kernels with block coordinate descent. In: International Conference on Machine Learning (ICML), pp. 49–56 (2011)
11. Evgeniou, T., Micchelli, C.A., Pontil, M.: Learning multiple tasks with kernel methods. J. Mach. Learn. Res. **6**, 615–637 (2005)
12. Gönen, M., Alpaydın, E.: Multiple kernel learning algorithms. J. Mach. Learn. Res. **12**, 2211–2268 (2011)
13. Günther, M., Klotz, L.: Schur's theorem for a block Hadamard product. Linear Algebra Appl. **437**(3), 948–956 (2012)
14. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel methods in machine learning. Ann. Stat. **36**, 1171–1220 (2008)
15. Huusari, R., Kadri, H., Capponi, C.: Multi-view metric learning in vector-valued kernel spaces. In: AISTATS (2018)
16. Kadri, H., Ayache, S., Capponi, C., Koço, S., Dupé, F.-X., Mor vant, E.: The multi-task learning view of multimodal data. In: Asian Conference in Machine Learning (ACML), pp. 261–276 (2013)

17. Kadri, H., Duflos, E., Preux, P., Canu, S., Rakotomamonjy, A., Audiffren, J.: Operator-valued kernels for learning from functional response data. J. Mach. Learn. Res. **16**, 1–54 (2016)
18. Kloft, M., Brefeld, U., Sonnenburg, S., Zien, A.: Lp-norm multiple kernel learning. J. Mach. Learn. Res. **12**, 953–997 (2011)
19. Koço, S., Capponi, C.: A boosting approach to multiview classification with cooperation. In: Proceedings of ECML PKDD'11, pp. 209–228 (2011)
20. Lampert, C.H., et al.: Kernel Methods in Computer Vision. Foundations and Trends® in Computer Graphics and Vision 4.3, pp. 193–285. Now Publishers, Delft (2009)
21. Lim, N., d'Alché Buc, F., Auliac, C., Michailidis, G.: Operator valued kernel-based vector autoregressive models for network inference. Mach. Learn. **99**(3), 489–513 (2015)
22. Maurer, A.: The Rademacher complexity of linear transformation classes. In: International Conference on Computational Learning Theory (COLT), pp. 65–78 (2006)
23. Micchelli, C.A., Pontil, M.: On learning vector-valued functions. Neural Comput. **17**, 177–204 (2005)
24. Minh, H.Q., Bazzani, L., Murino, V.: A unifying framework for vector-valued manifold regularization and multi-view learning. In: International Conference in Machine Learning (ICML) (2013)
25. Minh, H.Q., Bazzani, L., Murino, V.: A unifying framework in vector-valued reproducing kernel Hilbert spaces for manifold regularization and co-regularized multi-view learning. J. Mach. Learn. Res. **17**(25), 1–72 (2016)
26. Mohri, M., Rostamizadeh, A., Talwalkar, A.: Foundations of Machine Learning. MIT press, Cambridge (2012)
27. Pavlidis, P., Weston, J., Cai, J., Noble, W.S.: Learning gene functional classifications from multiple data types. J. Comput. Biol. **9**(2), 401–411 (2002)
28. Rudi, A., Camoriano, R., Rosasco, L.: Less is more: Nyström computational regularization. In: Advances in Neural Information Processing Systems (NIPS), pp. 1657–1665 (2015)
29. Sangnier, M., Fercoq, O., d'Alché Buc, F.: Joint quantile regression in vector-valued RKHSs. In: Advances in Neural Information Processing Systems (NIPS), pp. 3693–3701 (2016)
30. Schölkopf, B., Weston, J., Eskin, E., Leslie, C., Noble, W.S.: A kernel approach for learning from almost orthogonal patterns. In: European Conference on Machine Learning, pp. 511–528. Springer, Berlin (2002)
31. Sindhwani, V., Rosenberg, D.S.: An RKHS for multi-view learning and manifold co-regularization. In: International Conference in Machine Learning (ICML), pp. 976–983 (2008)
32. Sindhwani, V., Niyogi, P., Belkin, M.: A co-regularization approach to semi-supervised learning with multiple views. In: Proceedings of ICML Workshop on Learning with Multiple Views, pp. 74–79 (2005)
33. Sindhwani, V., Quang, M.H., Lozano, A.C.: Scalable matrix-valued kernel learning for high-dimensional nonlinear multivariate regression and granger causality. In: Uncertainty in Artificial Intelligence (UAI) (2012)
34. Weinberger, K.Q., Tesauro, G.: Metric learning for kernel regression. In: Artificial Intelligence and Statistics, pp. 612–619 (2007)
35. Williams, C.K.I., Seeger, M.: Using the Nyström method to speed up kernel machines. In: Advances in Neural Information Processing Systems (NIPS), pp. 682–688 (2001)
36. Xu, C., Tao, D., Xu, C.: A survey on multi-view learning (2013). ArXiv preprint arXiv:1304.5634
37. Zhang, Y., Yeung, D.-Y.: A convex formulation for learning task relationships in multi-task learning. In: Uncertainty in Artificial Intelligence (UAI) (2012)

Chapter 12
On the Evaluation of Community Detection Algorithms on Heterogeneous Social Media Data

Antonela Tommasel and Daniela Godoy

Abstract One fundamental problem in social networks is the identification of groups of elements (also known as communities) when group membership is not explicitly available. Community detection has proven to be valuable in diverse domains such as biology, social sciences and bibliometrics. Thus, several community detection techniques have been developed. Nonetheless, as real networks are very heterogenous, the question of how communities should be assessed remains open. Whilst there are several works that have analysed the performance of diverse community detection algorithms over artificial graph benchmarks, the evaluation over real social networks has received comparatively less attention. Motivated by the lack of such studies, this chapter focuses on the analysis of the performance of community detection algorithms over social media networks, and the quantification of the structural properties of the discovered communities.

12.1 Introduction

Social networking and microblogging sites have increased their popularity in recent years attracting millions of users, who spend an increasing amount of time sharing personal information and making new friends. For example, sites like *Flickr*, *YouTube*, *Facebook* or *Twitter* allow users to create content, publish photographies, comment on content other users shared, tag content and socially connect with other users in the form of subscriptions or friendships. Consequently, social networking sites affect how people communicate and interact, leading to the formation of relationships of heterogeneous nature, origin and strength. Users might choose their friends because they publish interesting information, share common interests or common friends, or just because they are celebrities, amongst other possible explanations. Thereby, topological relations could lead to the existence of

A. Tommasel (✉) · D. Godoy
ISISTAN, CONICET-UNICEN, Tandil, Buenos Aires, Argentina
e-mail: antonela.tommasel@isistan.unicen.edu.ar

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6_12

casual links. In this context, the significance and importance of relations should not be only analysed based on topological information, but in conjunction with other information sources or data views, which might implicitly define connections between social media users. For example, whether two users use the same terms or hashtags, or post on the same topics. It is worth noting that the content users consume or post might depend, for example, on their mood and environment [8].

One fundamental problem in social networks is the identification of groups of elements (users, posts or other elements) when group membership is not explicitly available. A group or community can be defined as a set of elements that interact more frequently or are more similar to other community members than to outsiders. Community detection has proven to be valuable in diverse domains such as biology, social sciences and bibliometrics. For example, community detection techniques can be used for identifying groups of users with similar purchase history on *Amazon* to create more efficient product recommendation systems, detecting topics in collaborative systems, identifying real-world landmarks in *Flickr* by clustering photos, detecting events on *Twitter* streams or for studying the information diffusion problem by solving the influence maximisation problem in *Foursquare*.

In the context of multidimensional networks, considering only one information source might be insufficient for accurately capturing community structure [40]. For example, in *Twitter*, social relations might be sparse, and users might belong to the same community even if there are no explicit friendship relations amongst them. Relations can also be noisy. As it is easier to connect with other users online than in the real world, users might have thousands of online friends. Hence, the correct identification of communities might be hindered if only friendship interactions are considered. Conversely, other users might have a few friends, but frequently engage in posting or commenting activities, which could reveal valuable information for discovering communities, despite the fact that social media content might be topically diverse and noisy. Thus, the integration of multiple information sources could help to overcome the problem caused by incomplete or noisy information in each dimension, as well as obtaining more accurate and reliable community partitions. Nonetheless, combining multiple and possibly heterogeneous data views poses new challenges; for example, how to fuse the different views for performing an integrated analysis.

Several community detection methods have been developed based on techniques from a variety of disciplines, such as statistical physics, biology, applied mathematics, computer science or sociology [44], mostly relying on similarity measurements amongst the nodes in the network [38], which might not be simple. Interestingly, most of them only focus on one data view. Moreover, although all methods aim at identifying meaningful communities, as they might rely on different notions of communities, their results might not be always directly comparable. In most real-world applications, a unique correspondence between nodes and communities (i.e. a ground truth) might not be available, which hinders the reliability assessment of community detection techniques. As a result, community detection algorithms are traditionally tested on a few real or artificial networks [30]. As real-world social networks are very heterogeneous, the question regarding over which data

evaluate the algorithms remains open. On the other hand, artificial networks rely on various statistics like average degree, degree distribution and shortest path average, amongst others, which are not possible to control in a real environment. Hence, in both cases, algorithms are usually evaluated over networks with very specific and limited set of characteristics, which might not match the typical features of real-world networks. Whilst there are several works that have analysed the performance of diverse community detection algorithms over artificial graph benchmarks [12, 18, 30, 44], their evaluation over real social networks has received comparatively less attention [20].

Considering the increasing amount of available information in social networks, the necessity of integrating such heterogeneous information and the lack of studies analysing the problem of community detection over real-world networks, this chapter addresses three challenges. First, the definition and extraction of multiple sources of information regarding user interactions and activities that can be inferred from social media data. Second, the assessment of the performance of community detection algorithms in the context of two real-world social media networks. Third, the exploration and evaluation of diverse similarity measures that could be considered during the community detection process. To that end, it is also explored how to quantify the structural properties of the discovered communities in terms of several quality metrics. The final goal of this study is to provide some insights regarding the integration of diverse information sources and user interactions, as well as the selection of both algorithms and metrics for performing and assessing the community detection process.

The rest of this chapter is organised as follows. Section 12.2 discusses related research. Section 12.3 describes background concepts regarding the definition and extraction of graphs from social media sites, and presents the community detection techniques and the similarity metrics evaluated in this study. Section 12.4 describes the experimental evaluation performed over two real-world networks from *Twitter* and *Flickr*. Section 12.5 presents the observed results. Finally, Sect. 12.6 summarises the results and conclusions drawn from the analysis.

12.2 Related Work

According to graph theory [22], communities have also been defined as cliques (every node is adjacent to each other) or connected components (every pair of nodes is connected by at least a path). In this context, the goal of community detection techniques (also known as graph clustering techniques) is to divide the nodes into communities (or clusters), such that the nodes of a particular community are similar or connected in some predefined sense [38]. Several works have been dedicated to formalise the intuition that a community is a set of nodes that has more or better connections between its members than with the remainder of the network [20]. For example, in some cases it might be desirable to obtain communities of similar order and/or density. Interestingly, not every graph presents a structure with natural

communities. In the case of a uniform graph structure in which edges are evenly distributed over the set of nodes, clustering results will be rather arbitrary.

Community detection techniques can be either local or global. Generally [31], the definition of global communities relies on the number of edges falling between them (cut size), the profoundness of their separation, modularity (i.e. the extent to which a given community partition deviates from the hypothetical state in which the network would be randomly rewired under the constraint of same-degree for each node [27]) or on the similarity between nodes. Community detection can be performed either by considering all data elements at once, or by iteratively assigning one element at a time to the appropriate cluster. Approaches that require the entire graph to be simultaneously accessible do not scale for large graphs [38].

On the other hand, local community detection techniques provide an alternative to alleviate scalability challenges of global techniques as they only focus on a portion of the network under study. Thus, they are expected to circumvent the memory bottleneck faced by global methods. Since it is not feasible to study the community structure as a whole in terms of space and computational complexity, communities can be progressively discovered by means of the explicit or implicit relations defined between the nodes. This type of technique starts the network exploration process from a set of seed nodes and progressively adds adjacent nodes to the community as long as those node additions lead to the increment of some local community quality measure [31].

In general, local techniques present limitations that need to be addressed in order to be effectively applied on large-scale social networks. First, the performance of local techniques is affected by the density (the number of links of the interconnected communities and the total number of links of the network) and the size of networks [29]. In this regard, techniques based on optimising modularity might fail to identify communities that are smaller than a size that depends on the number of nodes in the network and the link density of communities, even when communities are unambiguously discovered [13]. Fortunato and Barthélemy [13] found that the detection of communities based on modularity is not consistent with the modularity optimisation, which might favour network partitions in large communities. According to the authors, by enforcing modularity optimisation, the different possible partitions of the network are explored at a coarse level, so that communities that are smaller than a determined scale might not be resolved. The origin of the resolution scale lies in the fact that modularity is a sum of terms, where each term corresponds to a community. Thus, finding the maximal modularity is equivalent to look for the ideal trade-off between the number of terms in the sum. An increment in the number of communities does not necessarily imply an increment in modularity, as communities would be smaller so each term of the sum would also be smaller. Furthermore, modularity optimisation results in communities with similar sizes, which causes modularity to have a peak. The problem is that the supposedly optimal partition imposed by mathematics does not necessarily capture the actual community structure of the network, in which communities might have heterogeneous sizes. As a result, alternative measures for analysing community partitions have to be devised.

The remainder of this section presents relevant works that have aimed at defining benchmarks for comparing the performance of community detection techniques (Sect. 12.2.1), and metrics for comparing the performance of community detection techniques (Sect. 12.2.2).

12.2.1 Benchmarks for Community Detection

For newly designed techniques, it is necessary to assess their performance and compare it with that of other techniques. Nonetheless, the evaluation of techniques has received little attention in the literature [12]. As a result, it might be difficult to determine which technique is most reliable in the context of a certain domain or application. Generally, evaluations consist in applying the new techniques to a small set of simple benchmark graphs, whose community structure is known or easy to recover; for example, the social network of Zachary's karate club [45], the social network of bottlenose dolphins living in Doubtful Sound [21] or the American college football teams [14]. Zachary's karate club is one of the most used graphs and comprises two communities. In the American college football team graph, there are 115 nodes representing the teams, which are connected if they have played against each other. The natural partition of the graph comprises 12 communities, each representing a geographical area. Note that both networks are undirected and non-overlapping, as it is very difficult to find directed graph datasets with known community structures and sufficient size [22]. When considering real networks, it is worth noting that there is no guarantee that meaningful communities, defined on the basis of non-structural information, will match those detected by methods solely based on graph structure [12]. Moreover, in most cases, graph datasets are of small scale, which hinders their usefulness for assessing the performance of techniques at large scales. Thereby, it is crucial that the scientific community agrees on a standard evaluation procedure. In this context, several works have focused on the design of artificial benchmark graphs.

Condon and Karp [6] proposed one of the first graph benchmarks based on the planted ℓ-partition model with $n = g \cdot \ell$ nodes divided in ℓ communities with g nodes each. In this model, nodes of the same community are connected with a probability p_{in}, whilst nodes belonging to different communities are connected with a probability p_{out}. In this regard, each community represents a random Erdös-Réngy graph with a connection probability $p = p_{\text{in}}$. The modelled graph will have a community structure when the intracluster edge density is higher than the intercluster edge density, i.e. $p_{\text{in}} > p_{\text{out}}$. Following this model, Girvan and Newman [14] introduced one of the most known benchmarks, which is parametrised so that each network has 128 nodes divided into 4 groups, implying that p_{in} and p_{out} are not independent [12].

Although the ℓ-partition model is widely used, all nodes have approximately the same degree, and all communities have the same size by construction. These two features might not reflect the characteristics of real networks, in which degree

distributions might be skewed with many nodes with low degree coexisting with a few nodes with high degree (for example, in online social networks). Brandes et al. [3] proposed a modification to the model named Gaussian random partition generator in which community sizes have a Gaussian distribution. This variation on community sizes also introduces heterogeneity in the degree distribution, as the expected degree of a node will depend on the number of nodes in its community. However, the introduced variations might still not be enough to represent real networks. In all cases, the hypothesis that the connecting probabilities of each node with the other nodes in the community or even with other communities are constant might not represent the actual properties of networks.

Lancichinetti et al. [19] proposed the LFR benchmark, in which the distributions of degree and community sizes are assumed to be governed by independent power laws. Graphs are built as follows. First, community sizes are defined based on a predefined power law distribution. Second, each node in a community is assigned a degree, which is defined considering another predefined power law. Third, all stubs of vertices of the same community are randomly connected to each other to maintain the predefined degree distribution. Fourth, each node is connected to nodes in the other communities.

It is worth noting that the described benchmarks correspond to undirected networks. Although the problem of detecting communities on directed networks has received comparatively less attention than on undirected networks, several benchmarks have been introduced to deal with special types of graphs and community structures. For example, Arenas et al. [1] extended Girvan and Newman's [14] benchmark to build graphs with embedded hierarchical structures. Additionally, Lancichinetti and Fortunato [18] proposed a modification of the LFR benchmark to create weighted, directed, and unweighted and overlapping networks. The weighted graph is built based on an unweighted graph by assigning positive real numbers to each edge. To that end, two new parameters are defined. The first parameter is used to assign a strength to each node such that the power law relation between the strength and the node degree is frequently observed in real networks. The second parameter is used to assign the internal strength, which is defined as the sum of the weights of the connections between a node and all its neighbours belonging to the same community. Then, a greedy algorithm is applied to the graph so that weights are consistent with the connection probabilities. For directed networks, changes are imposed in the degree definition. Whilst in-degrees are defined based on a power law, out-degrees are defined based on a δ-distribution. Finally, connections are established by preserving both distributions.

In the case of unweighted and overlapping graphs, a new topological mixing parameter is introduced to define the number of neighbours of a node that have at least one membership in common. The generating procedure is equivalent to the generation of a bipartite network where the two classes are the communities, and nodes comply to the requirement that the sum of community sizes equals the sum of node memberships. Although this benchmark aims at simulating the features observed in real-world networks, the requirement that overlapping nodes interact with the same number of embedded communities might be unrealistic [42]. Simple

generalisations were proposed in which each overlapping node might belong to different numbers of communities [25], or communities are converted to fuzzy associations by adding a belonging coefficient to the occurrence of nodes [15]. Similarly, Sawardecker et al. [37] also proposed an extension of Girvan and Newman [14]'s benchmark in which the probability of an edge to be present in the network is a non-decreasing function based on the set of co-memberships of its vertices.

12.2.2 Comparing Community Detection Techniques

In addition to considering graphs with a known community structure, the quality of the communities discovered by a technique should be compared to that of other techniques, aiming at selecting the most accurate one. This implies the selection of criteria depending on the known structure of the network to define how similar the discovered communities are. Several metrics have been used for this purpose, which can be divided into three categories [12]: metrics based on pair counting, community matching or information theory. Metrics based on pair counting depend on the number of pairs of nodes that are assigned to the same or different communities. Similarity metrics based on community matching aim at finding the largest overlap between pairs of communities belonging to different partitions. A common problem of this type of metrics is that some communities might not be considered if the overlap with other communities is not large enough. The third category is based on casting the problem of comparing communities as the problem of message decoding in the context of information theory. The rationale of these metrics is that if two community partitions are similar, little information is needed to infer one partition, given the other. Thus, such extra information can be used as a measure of dissimilarity. Evaluating the quality of the discovered communities is non-trivial. The problem worsens when analysing overlapping communities [42], as extending the evaluation metrics from disjoint to overlapping communities is rarely straightforward.

Generally, there are two criteria to analyse the goodness of a community partition [20]. First, the number of edges or links amongst the nodes in each community, and second, the number of edges amongst the members of each community in relation to the nodes outside of it. These criteria characterise the connectivity structure of a given community, built on the assumption that communities comprise sets of nodes with many inner connections and few outer connections. Table 12.1 presents the definitions of the used metrics, where S represents the set of nodes of a community, E represents the set of edges of a community, n_S is the number of nodes in community S, c_S is the number of edges in the boundary of the community, i.e. $c_S = |\{(u, v) : u \in S, v \notin S\}|$, m_S is the number of edges inside the community, i.e. $m_S = |\{(u, v) : u, v \in S\}|$, $d(u)$ is the degree of node u, σ_{sw} is the number of shortest paths from node s to w and $\sigma_{sw}(v)$ is the number of shortest paths from

Table 12.1 Community quality measures

Cut ratio (↓)	Measures the fraction of existing edges (out of all possible edges) leaving the community. $\frac{c_S}{n_S(n-n_S)}$
Triangle separation ratio (TPR) (↑)	Measures the fraction of nodes inside a community that belong to a triad. $1/n_S * \lvert\{u : u \in S, \{(v,w) : v, w \in S, (v,w) \in E, (u,w) \in E, (v,w) \in E\} \neq \varnothing\}\rvert$
Conductance (↓)	Measures the fraction of total edge volume that points outside the community. $\frac{c_S}{2 * m_S + c_S}$
Flake out degree fraction (FlakeODF) (↓)	Measures the fraction of nodes in the community that has fewer edges pointing inside the community than outside. $1/n_S * \lvert\{u : u \in S\}, \lvert\{(u,v) \in E, v \in S\}\rvert < d(u)/2\rvert$
Betweenness centrality (↑)	Measures how often a node appears on shortest paths between nodes in the community. $1/n_S * \sum_{u,v,w \in S, u \neq v \neq w} \frac{\sigma_{sw}(v)}{\sigma_{sw}}$
Closeness centrality (↑)	Measures the average distance between every pair of nodes in the community. $1/n_S * \sum_{u,v \in S, u \neq v} \frac{1}{\text{distance}(u,v)}$
Eccentricity (↓)	Averages the distance from each node to the farthest node in the community. $1/n_S * \sum_{u \in S} \max\{\text{distance}(u,v) : v \in S\}$
Density (↑)	Measures the fraction of edges (out of all possible edges) that appear between the nodes in the community. It is based on the supposition that good communities are well connected. $\frac{2 * m_S}{n_S(n_S-1)}$
Clustering coefficient (↑)	It is based on the supposition that communities are manifestations of locally inhomogeneous distributions of edges as pairs of nodes with common neighbours are more likely to be connected with each other. $1/n_S * \sum_{u \in S} \frac{\lvert(v,w) \in E : v, w \in S, (u,v) \in E \wedge (u,w) \in E\rvert}{d(u) * (d(u)-1)}$
Separability (↑)	Measures the ratio between the internal and external number of edges in the community. It is based on the supposition that good communities are well-separated from the rest of the network, i.e. communities have relatively few edges pointing to other communities. $\frac{m_S}{c_S}$

node s to w that pass through v. The arrows indicate whether a higher (↑) or a lower (↓) score is preferable.

Considering the described types of metrics, Lancichinetti and Fortunato [18] analysed the performance of 12 algorithms in the context of undirected and unweighted, and directed and weighted networks. Performance was evaluated in terms of computational time and normalised mutual information. Yang et al. [44] evaluated 8 algorithms by quantifying their accuracy using complementary measures and their computing time. The authors aimed at studying the dependency between network size, computing time and the predicting power of techniques. Finally, closely related to this study, Leskovec et al. [20] evaluated community detection algorithms over real networks: a bipartite authors-papers networks from

DBLP,[1] the Enron e-mail network,[2] a co-authorship network from Arxiv[3] and a social network from Epinions.[4] Note that no social media data network was selected for the analysis. The authors aimed at understanding the biases in the communities discovered by the selected algorithms by analysing several objective functions.

12.3 Community Detection on Social Media

Considering the increasing amount of available information in social networks, the development of a large number of community detection techniques, the necessity of integrating heterogeneous data from multiple source, and the lack of studies analysing either the performance assessment of community detection techniques, or the problem of community detection over real-world networks, this chapter focuses on four aspects. First, the definition of a graph in the context of social media data (Sect. 12.3.1). Second, the analysis of the performance of several community detection techniques (presented in Sect. 12.3.2). Third, the exploration and evaluation of diverse similarity metrics that could be considered during the community detection process (defined in Sect. 12.3.3). Fourth, the quantification of the properties and characteristics of communities that determine the goodness of algorithms (presented in Sect. 12.3.4). The final goal of this study is to provide some insights regarding the integration of diverse information sources and user interactions for community detection, as well as the selection of both the algorithms and the metrics for assessing the community detection process.

12.3.1 Graph Derivation on Social Media

To apply a community detection algorithm, the information on which the underlying graph structure is going to be based on has to be defined. Multiple and diverse information sources can be extracted from social media data, and hence multiple graph structures can be defined. Nodes might not only represent real people but also other entities such as neighbourhoods, Web pages or tweets, amongst others, depending on the task at hand to perform [23]. Then, once communities are found, they can be integrated in diverse learning tasks such as topic detection, text classification or clustering, link prediction or even feature selection.

Most community detection techniques are purely based on explicit social relations; however, in the context of social media data, both the social relations between

[1] http://snap.stanford.edu/data/com-DBLP.html.

[2] http://snap.stanford.edu/data/email-Enron.html.

[3] http://snap.stanford.edu/data/cit-HepPh.html.

[4] http://snap.stanford.edu/data/soc-Epinions1.html.

users and the characteristics of the published content are important for improving the quality of the discovered communities [40]. Hence, besides the relations between posts derived from the actual social relations between their authors (i.e. two posts are socially connected if their authors are socially related), posts' content resemblance or common categories (in case they are available) could also help to establish relations between them. It is worth noting that social information and content-based relations offer complementary views of data, in this case, posts. Thus, no individual relation might be sufficient for accurately determining community memberships [39]. For example, social information might be sparse and noisy, whilst content-based information could be irrelevant or redundant. Hence, it is important to adequately combine the different types of relations for performing community detection in social networks.

Content-based relations could be used either to establish new relations between posts that are not socially related (named *Independent* graph derivation) or to reinforce the social relations already found amongst posts (named *Weighted* graph derivation). In the former case, social and content relations are assumed to be independent from each other, i.e. edges in the graph represent not only social links but also separated content ones. Hence, when considering both types of relations independently, two nodes might be connected even when there is no explicit social connection between them. In this graph derivation, the different relationships are integrated by adding their corresponding matrices, as $A_{Rels} = \sum_{i \in Rels} A_i$, where A_{Rels} represents the aggregated adjacency matrix, $Rels$ is the set of selected relationships and A_i are the adjacency matrices. Note that no differentiation is made between the social and content-based relationships.

On the *Weighted* derivation, the graph only includes edges representing the social relation between nodes, whose strength or relevance is given by the content features. Thus, in this case, the quality of the social ties between nodes depends on an adequate definition of the content-based features, which should allow to fully exploit the social media data information. The final adjacency matrix for this derivation can be defined as $A_{Rels} = A_{Social} \cdot \sum_{i \in Rels_W} A_i + \sum_{i \in \{Rels - Social - Rels_W\}} A_i$, where A_{Social} represents the adjacency matrix for the *Social* relation and $Rels_W$ the set of relationships chosen for weighting the *Social* relationship. Note that this graph derivation also allows the integration of independent relationships, as showed by the second term in the Equation.

By definition, all content-based relations are symmetric, i.e. they do not have directionality. However, the same does not necessarily apply to the social or friendship relations. For example, when considering the Follower/Followee relationship in *Twitter* or the social relations in *Instagram*, the fact that user A follows user B does not imply that user B also follows user A. Whilst the diverse social networks exhibit different reciprocity levels, most community detection techniques only leverage on undirected (and perhaps weighted) graphs. It is worth noting that developing community detection techniques for directed graphs might be a difficult task [12], and that several concepts that are theoretically well defined for undirected graphs have not been yet extended to directed ones [22].

12.3.2 Algorithms to Compare

Both global and local community detection algorithms were selected for the comparison, which are described as follows.

Cobweb It is an incremental clustering algorithm [11]. The goal is to learn from observations, in opposition to learning from examples. Cobweb incrementally organises observations into a classification tree, in which each node is labelled by a probabilistic concept that summarises the attribute-value distributions of the elements under such particular node. Search is guided by a heuristic measure called Category Utility, which can be regarded as a trade-off between the intraclass similarity and interclass dissimilarity of elements.

Edge Betweenness It is a hierarchical decomposition process in which edges are removed in the decreasing order of their Edge Betweenness score [14, 27]. This technique focuses on the least central edges that connect separated communities. Hence, communities are created by progressively removing edges from the original graph. The computational complexity of the algorithm is $\Theta\left(v \cdot e^2\right)$, where v represents the number of nodes in the graph and e the number of edges, which might make it impractical for large and dense graphs. One disadvantage of this algorithm is that it builds a full dendrogram and does not provide any guidance about where to cut it to obtain the final community partition, hence other measures are needed to find that optimal partition.

Expectation Maximisation (EM) It is an iterative algorithm that alternates between two steps, expectation (E) and maximisation (M) [9]. When applied to clustering, expectation maximisation (EM) uses finite Gaussian mixture models and iteratively estimates a set of parameters until a desired convergence value is achieved. In the E step, for each instance it computes its membership possibility to each cluster based on the initial parameters. In the M step, parameters are recomputed based on the new membership possibilities. Theoretically, the running time is not bounded.

Farthest First It is a variant of the K-means algorithm that places each cluster centre aiming at maximising the cluster radius [16]. The algorithm operates in two steps: centroid selection and cluster assignment. The centroid selection step begins by selecting a random data point as the original cluster centre. Then, it iteratively chooses the next centres as the data points that are farthest from the previously selected one, until the desired number of centroids has been selected. In the cluster assignment step, all other data points are assigned to the nearest centroid. The computational complexity of this algorithm is $\Theta\left(v \cdot k\right)$, where k represents the number of desired clusters, making it suitable for large-scale applications.

Fast Greedy It is a bottom-up hierarchical approach that tries to optimise modularity in a greedy manner [5]. At first, each node belongs to a separated community. Then, communities are iteratively merged such that each merge is locally optimal. The merges stop when it is infeasible to increase modularity. The computational

complexity of the algorithm is $\Theta(e \cdot d \cdot \log(v))$, where d represents the depth of the dendrogram describing the community structure. This algorithm faces the resolution limit problem, caused by inefficiencies derived from merging communities with unbalanced sizes [41].

Infomap This algorithm optimises the map equation [36] exploiting the information-theoretic duality between the problem of data compression, and the problem of how to extract significant patterns or structures from such data. The map equation specifies the theoretical limit of the trajectories of a random walker on the network. Community structure is represented through a two-level nomenclature based on Huffman coding. The rationale behind this optimisation is that on partitions containing few intercommunity paths, the walker will probably stay longer inside the communities, leading to compact representations. The computational complexity of the algorithm is $\Theta(e)$.

Label Propagation In this algorithm [34], each node is assigned one of k labels. Then, labels are iteratively reassigned such that nodes take the most frequent label of its neighbours synchronously. The process is repeated until each node is labelled with the most frequent label in its neighbourhood. This algorithm is solely based on network structure and does not require neither optimisation of an objective function nor prior information about the communities. Although it is a fast technique, it provides unstable results, as they depend on the initial label configuration and the random decision of breaking ties. The computational complexity of the algorithm is $\Theta(v + e)$.

Leading Eigenvector It is a top-down hierarchical approach that optimises modularity in terms of a matrix eigenspectrum [26], leading to a centrality measure that identifies those vertices that occupy central positions within the communities to which they belong. The algorithm starts by computing the leading eigenvector of the modularity matrix. Then, it iteratively splits the graph into two parts to maximise the modularity improvement based on the leading eigenvector and stops once the modularity of the network subdivision is negative. The running time of the algorithm is $\Theta\left(v^2 + v \cdot e\right)$, or $\Theta\left(v^2\right)$ for sparse graphs.

Louvain Algorithm It is one of the most known and easy to implement algorithms, based on a greedy optimisation of modularity [2]. The algorithm is divided into two steps that are iteratively performed until there are no more changes. First, each node in the graph is assigned to a different community. For each node, it is moved to the community for which the positive modularity gain is maximum. This process is sequentially and repeatedly applied until modularity cannot be improved. The second step builds a new network whose nodes are the communities found during the first step. By definition, the number of communities decreases at each pass, thus most of the running time is concentrated on the first iterations. The computational time of the algorithm is $\Theta(v \cdot \log(v))$.

Spinglass This algorithm aims at finding communities via a Spinglass model and simulated annealing based on statistical physics [35]. In this model, each particle

(i.e. node) can be in one of c spin states (i.e. the maximum number of communities), and the interactions amongst particles (i.e. edges between nodes) define which nodes might stay on the same state, and which are likely to have different states. Then, the model is simulated and the final spin states of particles define the final community partition. Note that spin states could remain empty, hence reducing the number of found communities. Due to the nature of simulations, the algorithm is not deterministic, but it has parameters that allow determining the sizes of the communities to be found. The computational complexity of the algorithm for sparse graphs is approximately $\Theta\left(v^{3.2}\right)$.

Walktrap Similarly to Infomap, this algorithm is based on random walks [33]. The rationale is that short distance random walks tend to stay in the same community as they are assumed to have few edges outside them. At first, all nodes belong to different communities, and the distances between all adjacent nodes are computed. Then, two adjacent communities are iteratively chosen and merged, updating the corresponding distances. The output of the algorithm is a full dendrogram. The computational complexity of the algorithm is $\Theta\left(v^2 \cdot e\right)$, and $\Theta\left(v^2 \cdot \log\left(v\right)\right)$ for sparse graphs.

X-Means This algorithm aims at overcoming three K-means shortcomings [32]: the scaling inefficiency, the manual definition of the number of clusters, and the proneness to remain on local minima. It works as K-means until all instances have been assigned to their corresponding community or cluster. Then, it attempts to split each community into two separate communities by computing the Bayesian information criterion (BIC) on both the original community and the two newly created ones. In those cases the BIC for the new communities is higher than that for the previously defined communities, the new communities are retained and the total number of communities is increased. This process is iteratively repeated until convergence. Once all K values are tested, the best community partition is chosen.

12.3.3 Vertex Similarity

Most community detection techniques are based on computing the similarity amongst nodes [38]. For example, in a graph in which nodes represent posts, the similarities amongst them could be used to group together nodes that are not only well connected but also similar to each other. However, assessing node similarity might not be computationally simple, and even more complex than clustering the graph once all similarities are known.

The quality of the discovered communities might be greatly affected by the selection of the similarity metric. Hence, such metric has to be carefully chosen. Furthermore, as the different metrics assess node similarity from different points of view, they could be combined to perform a more comprehensive assessment. The Harmonic mean, which is less biased to the presence of outliers than the Arithmetic

mean, adds a possibility for combining similarity scores. Note that for combining the scores, they must be normalised to the same range.

Metrics for computing similarity amongst nodes can be divided into three groups [38], which are summarised in Table 12.2:

Distance/Similarity Metrics Traditionally, these metrics are based on some distance property. Distance metrics should comply with three criteria: the distance between a node and itself is zero, distances are symmetrical and the triangle inequality holds. Similarity metrics resulting from the adaptation of distance metrics also comply with the three criteria. A great number of distance metrics have been defined and used in the literature [10, 17]. Examples of these metrics include Euclidean Distance, Manhattan Distance and Tanimoto Coefficient, amongst others.

Adjacency-Based Metrics In several environments, nodes lack of properties that allow computing their similarity [38]. The most straightforward manner for determining whether two nodes are similar only using adjacency information is to analyse the overlap of their neighbourhoods. Nonetheless, correlation analyses could also be applied to determine community structure based on adjacency information. Examples of these metrics include [10, 17]: Jaccard Similarity, Common Neighbours and Pearson Correlation, amongst others.

Connectivity Metrics Communities in graphs can also be defined through node connectivity by computing the number of paths between each pair of nodes [38]. In this regard, nodes belonging to the same community should be highly connected to each other. Connectivity metrics could be used to define the similarity amongst nodes. For example, nodes could be regarded as similar if they are connected by a number of paths higher than a predefined threshold, or similarity could be defined proportionally to the number of paths connecting the nodes. However, defining the threshold might be difficult, as its selection often involves knowing the diameter of the graph a priori. Choosing a large threshold in relation to the diameter of the graph might result in communities containing large portions of the graph, whereas choosing a small threshold might split natural communities into two or more subcommunities.

12.3.4 Quality Metrics

The criteria proposed by Leskovec et al. [20] (summarised in Table 12.1) characterise the connectivity structure of a given community built on the assumption that communities comprise sets of nodes with many inner connections and few outer connections. Nonetheless, connectivity structure might not be the only important characteristic of communities. In this regard, two additional functions were considered. First, a function characterising communities' content cohesiveness defined as the average Cosine Similarity amongst all node pairs in the community (named *ContentCohesiveness*). Second, as *Twitter* trending topics and *Flickr* photos

Table 12.2 Vertex similarity metrics

Statistic-based	Pearson Correlation (p_i, p_j)	$\frac{n\left(\sum_{k=1}^{n} A_{p_i,p_k} * A_{p_j,p_j}\right) - \lvert\Gamma(p_i)\rvert * \lvert\Gamma(p_j)\rvert}{\sqrt{\lvert\Gamma(p_i)\rvert * \lvert\Gamma(p_j)\rvert\,(n - \lvert\Gamma(p_i)\rvert) * (n - \lvert\Gamma(p_j)\rvert)}}$
Adjacency-based	$Sørensen\,(p_i, p_j)$	$\frac{2\lvert\Gamma(p_i) \cap \Gamma(p_j)\rvert}{\lvert\Gamma(p_i)\rvert + \lvert\Gamma(p_j)\rvert}$
Adjacency-based	$Leicht - Holme - Newman - LHN\,(p_i, p_j)$	$\frac{\lvert\Gamma(p_i) \cap \Gamma(p_j)\rvert}{\lvert\Gamma(p_i)\rvert \times \lvert\Gamma(p_j)\rvert}$
Adjacency-based	$Hub\ Promoted\ Index - HPI\,(p_i, p_j)$	$\frac{\lvert\Gamma(p_i) \cap \Gamma(p_j)\rvert}{\min\{\lvert\Gamma(p_i)\rvert, \lvert\Gamma(p_j)\rvert\}}$
Adjacency-based	$Hub\ Depressed\ Index - HDI\,(p_i, p_j)$	$\frac{\lvert\Gamma(p_i) \cap \Gamma(p_j)\rvert}{\max\{\lvert\Gamma(p_i)\rvert, \lvert\Gamma(p_j)\rvert\}}$
Adjacency-based	$Jaccard\,(p_i, p_j)$	$1 - \frac{\lvert\Gamma(p_i)\cap\Gamma(p_j)\rvert}{\lvert\Gamma(p_i)\cup\Gamma(p_j)\rvert} = \frac{\lvert\Gamma(p_i)\cup\Gamma(p_j)\rvert - \lvert\Gamma(p_i)\cap\Gamma(p_j)\rvert}{\lvert\Gamma(p_i)\cup\Gamma(p_j)\rvert}$
Distance-Similarity	$Euclidean\,(p_i, p_j)$	$\sqrt{\sum_{t=0}^{m}\left(p_{i,t} - p_{j,t}\right)^2}$
Distance-Similarity	$Manhattan\,(p_i, p_j)$	$\sqrt{\sum_{t=0}^{m}\lvert p_{i,t} - p_{j,t}\rvert}$
Distance-Similarity	$Tanimoto\,(p_i, p_j)$	$\frac{p_i \cdot p_j}{\lVert p_i\rVert^2 + \lVert p_j\rVert^2 - p_i \cdot p_j}$
Distance-Similarity	$Cosine\,(p_i, p_j)$	$\frac{p_i \cdot p_j}{\lVert p_j\rVert\,\lVert p_j\rVert} = \frac{\sum_{t=0}^{m}\left(p_{i,t} \times p_{j,t}\right)}{\sum_{t=0}^{m} p_{i,t}^2 \times \sum_{t=0}^{m} p_{j,t}^2}$

Where p_i and p_j denote the post for which the similarity score is computed, $\Gamma(p_i)$ denotes the set of neighbours of p_i, $|\Gamma(p_i)|$ denotes the degree of post p_i, A denotes the adjacency matrix, n denotes the number of posts contained in the graph and m denotes the dimension of posts, i.e. the total number of features contained in all posts

are assigned to classes, the *Entropy* of the classes given the community assignments was also analysed. As scores were computed for each individual community, they were averaged to obtain the score corresponding to a given community partition. To ensure metrics' comparability, all results were normalised to the range [0; 1], and adjusted so that the highest scores represent the best ones.

12.4 Experimental Evaluation

This section presents the experimental evaluation performed to assess the effectiveness of the selected community detection algorithms over social media data, and is organised as follows. First, Sect. 12.4.1 describes the data collections used. Then, Sect. 12.4.2 describes the implementation details. Finally, Sect. 12.4.3 details the graphs derived from social media used to perform the evaluation. The final goal of this study is to provide some insights regarding the selection of both the algorithms and the metrics for performing and assessing the community detection process.

12.4.1 Dataset

The performance of the technique was evaluated considering two real-world datasets collected from *Twitter*[5] [46] and *Flickr*[6] [24]. Table 12.3 summarises the main characteristics of both datasets. The *Twitter* dataset includes the content of more than 500,000 tweets belonging to 1036 trending topics, which were manually assigned to one of four categories: News, Ongoing Events, Memes (trending topics triggered by viral ideas) and Commemoratives (the commemoration of a certain person or event that is being remembered in a given day, for example birthdays or memorials). For the purpose of the experimental evaluation, each trending topic was regarded as a node in the graph, i.e. each node grouped the tweet set associated to the corresponding trending topic.

The *Flickr* dataset comprises the metadata associated to original images from the NUS-WIDE dataset [4]. For each photo, the dataset included information regarding the owner, description, title, comments, tags, manually annotated labels and the groups in which the photo was posted. Labels were considered as the category of photos, and hence the community ground truth. Photos could be assigned to 81 different concepts (belonging to different categories such as, scene, object, event, program, people and graphics), which were extracted from frequently used tags, representing either general (e.g. "animal") or specific (e.g. "dog") concepts. Only photos containing at least one tag or description were kept. The dataset also provides

[5] http://www.twitter.com/

[6] http://snap.stanford.edu/data/web-flickr.html.

Table 12.3 Data collection's main characteristics

(a) Twitter data collection	
Number of instances	1036
Number of features	226,043
Number of classes	4
Number of following relations	251,522,840
Average number of followees	816
Average number of features per instance	1084
Average number of instances per class	259
(b) *Flickr* data collection	
Number of instances	190,339
Number of features	947,829
Number of classes	81
Number of taggers	58,144
Number of commenters	569,765
Pairs of photos posted by the same user	77,909
Pairs of photos posted by users who are friends	8,825,738
Average number of features per instance	5
Average number of instances per class	1007

information regarding the topological relations between the users and their photos, including an indicator for whether both photos were taken by the same user, and an indicator for whether two users were socially related. For the purpose of the experimental evaluation, each photo was considered as a node in the graph.

12.4.2 Experimental Settings

The performance of the selected community detection and clustering techniques was evaluated for different graph sizes ranging between 50 and 1000 posts. In order to make the results comparable, the implementations of the algorithms provided by three widely used libraries were used: Gephi Toolkit,[7] WEKA[8] and Igraph.[9] Both Gephi Toolkit and WEKA are implemented in Java, whilst Igraph is available in R, C++ and Python. For the purpose of this evaluation, the Python implementation was

[7] https://gephi.org.

[8] https://www.cs.waikato.ac.nz/ml/weka/.

[9] http://igraph.org/.

chosen. For Cobweb, EM, Farthest First and X-Means clustering algorithms, their WEKA implementation was used. Specifically, the density-based implementations were selected to evaluate the generated partitions in a cross-validated manner. The graph was represented in the *arff* format in which each node was considered an instance, and features also represented nodes. The value of each feature represented the weight of the edge between the corresponding nodes. On the other hand, for Edge Betweenness, Fast Greedy, Infomap, Label Propagation, Spinglass and Walktrap, it was used their Igraph implementation. Finally, as regards the Louvain algorithm, the Gephi implementation was used. The number of communities or clusters to detect was automatically discovered in all cases. For those algorithms returning a full dendrogram, the chosen community partition corresponded to the one maximising modularity.

12.4.3 Graph Creation

Evaluation was performed considering the *Social* relationship and the combinations of relationships obtaining the best results in [40], regarding the independent and weighted graph derivations, which are summarised in Tables 12.4 and 12.5 for the *Twitter* and *Flickr* datasets, respectively. Additionally, the table shows for each of the analysed relationships the average and standard deviation of the node's degrees for each of the graph sizes tested. As most algorithms are designed to work with undirected graphs, a symmetrisation strategy was applied to the directed graphs resulting from considering the asymmetrical *Social* relationship. Particularly, a simple symmetrisation in which the new adjacency matrix U can be defined as $U = A + A^T$ was applied. As a result, in the case a pair of nodes is connected with edges in both directions, the weight of the edge in the symmetrised graph will correspond to the sum of the weight of the directed edges.

12.5 Experimental Results

Interestingly, not every tested algorithm was able to find a meaningful number of communities (i.e. a number between 1 and the number of nodes) for each of the analysed relationships, hence those results are not reported. Considering the results obtained for each of the evaluated graph sizes, it was analysed whether the different samples achieved similar quality; in other words, whether the algorithms behaved stable across different graph sizes. Data normality was evaluated by performing both the Shapiro and the Anderson–Darling tests [7]. As data was shown not to be normal, the Kruskal–Wallis test for unrelated samples was applied to the results obtained for each metric and analysed combinations of relations. Particularly, each of the results obtained for a determined graph size was regarded as a sample. The confidence value was set to 0.01. To perform the test, the null and the alternative hypothesis were

Table 12.4 Evaluated combinations of social and content-based relationships for the *Twitter* dataset

	50 nodes	100 nodes	200 nodes	500 nodes	1000 nodes
Independent graph derivation					
Social	36.80 ± 8.21	79.94 ± 14.62	153.30 ± 36.27	383.74 ± 89.20	763.96 ± 179.79
SimilarContent-0.6	0.08 ± 0.27	0.06 ± 0.24	0.02 ± 0.14	0.10 ± 0.33	0.21 ± 0.55
SharedClass	18.88 ± 7.72	43.72 ± 21.13	80.20 ± 38.21	195.57 ± 94.09	407.47 ± 193.51
Social & SimilarContent-0.6	36.84 ± 8.22	79.94 ± 14.62	153.30 ± 36.27	383.74 ± 89.20	763.96 ± 179.79
Social & SharedClass	40.68 ± 6.20	87.82 ± 11.21	171.22 ± 26.02	425.15 ± 66.32	854.87 ± 129.11
Social & SharedClass & SharedTag & SimilarContent-0.6	42.08 ± 5.43	89.19 ± 10.16	176.00 ± 23.87	435.63 ± 61.77	875.18 ± 119.09
Social & SharedClass & SharedTag & SimilarContent	49.00 ± 0.00	98.00 ± 0.00	198.76 ± 1.24	496.44 ± 11.91	993.19 ± 16.88
Weighted graph derivation					
Social-W-SimilarContent-0.6	0.04 ± 0.20	0.06 ± 0.24	0.02 ± 0.14	0.09 ± 0.33	0.21 ± 0.55
Social-W-SharedClass	15.00 ± 7.24	35.84 ± 17.87	62.28 ± 32.41	154.15 ± 83.49	316.56 ± 168.83
Social-W-SimilarContent-0.6 & SharedClass	18.88 ± 7.72	43.74 ± 21.13	80.20 ± 38.21	0.09 ± 0.33	407.50 ± 193.49
Social-W-SharedClass & SimilarContent-0.6	15.04 ± 7.27	35.86 ± 17.88	62.28 ± 32.41	154.16 ± 83.49	316.59 ± 168.81
Social-W-SharedTag & SharedClass	27.24 ± 8.84	60.67 ± 19.61	125.82 ± 38.56	296.62 ± 94.60	606.40 ± 188.85

Table 12.5 Evaluated combinations of social and content-based relationships for the *Flickr* dataset

	50 nodes	100 nodes	200 nodes	500 nodes	1000 nodes
Independent graph derivation					
Social	29.00 ± 3.41	72.50 ± 8.51	122.25 ± 17.88	299.73 ± 17.88	371.42 ± 78.85
SimilarContent-0.6	16.62 ± 2.14	36.93 ± 4.51	73.06 ± 8.02	176.88 ± 8.02	349.88 ± 16.51
TaggedSameUser	57.59 ± 3.58	363.12 ± 34.31	418.04 ± 26.05	486.29 ± 26.05	618.71 ± 35.11
Social & SimilarContent-0.6	24.93 ± 3.22	55.40 ± 6.77	109.60 ± 12.03	265.34 ± 12.03	524.87 ± 24.75
Social & TaggedSameUser	90.13 ± 7.52	376.35 ± 33.66	475.95 ± 27.24	635.66 ± 27.24	857.08 ± 35.82
Weighted graph derivation					
Social-W-SharedClass & SimilarContent-0.6	19.94 ± 2.57	44.32 ± 5.42	87.67 ± 9.62	212.25 ± 9.62	419.86 ± 19.81
Social-W-SimilarContent & SimilarContent-0.6	79.65 ± 11.29	226.40 ± 50.50	271.67 ± 40.86	240.33 ± 40.86	489.21 ± 53.73
Social-W-TaggedSameUser & SimilarContent-0.6	16.62 ± 2.14	36.93 ± 4.51	73.06 ± 8.02	176.88 ± 8.02	349.88 ± 16.51
Social-W-CommentedSameUser & SimilarContent-0.6	18.28 ± 2.36	40.63 ± 4.96	80.37 ± 8.82	194.57 ± 8.82	384.87 ± 18.15
Social-W-SharedTag & SimilarContent-0.6	19.11 ± 2.47	42.47 ± 5.19	84.02 ± 9.22	203.41 ± 9.22	402.36 ± 18.98

defined. The null hypothesis stated that no difference existed amongst the results of the different samples, i.e. the alternatives behaved stable over the different graph sizes. On the contrary, the alternative hypothesis stated that changes in the size of the graph caused changes in the behaviour of the algorithms. In all cases, the null hypothesis could not be rejected, due to a p-value higher than the confidence value, or a critical value higher than the obtained statistic value. In consequence, it could be assumed that the algorithms did not change their behaviour as the size of the graph changed. Hence, for clarity of presentation, the results across different graph sizes are summarised by their mean value. The statistical invariability of results confirms the findings in [44] that stated that the performance of the algorithms on artificial networks is independent of the network size.

12.5.1 Evaluation of Quality Metrics

A correlation test was applied to the results obtained for all metrics to assess the relationship between them. For all the obtained community structures, the scores corresponding to all the metrics in Table 12.1 were computed. The correlation between the metrics' results was evaluated according to the definitions and methods proposed in [7]. The normality of results was evaluated by analysing their skewness, kurtosis, and performing both the Shapiro and the Anderson–Darling tests. As the normality tests failed for at least one result sample, correlation results had to be evaluated by means of non-parametric tests. Thus, correlation was measured by means of the Spearman Rank Order correlation. The confidence value for considering a correlation statistically significant was set to 0.05. The minimum correlation value for two metrics to be considered highly correlated was set to 0.7. Figure 12.1 summarises the obtained results for the *Twitter* dataset by depicting the significant relations found. Interestingly, although *TPR* is not highly correlated with any other metric (the highest correlation was 0.52 with the *ClusteringCoefficient*), the obtained results showed that the metric has no sufficient discriminative power amongst the different evaluated community detection alternatives. Particularly, its standard deviation was 0.27. Consequently, *TPR* was not considered for assessing the quality of communities. As the figure shows, metrics can be grouped into four clusters, out of which a representative metric can be chosen. These results are in agreement with those in [43], showing that although there are different quality measures for structurally assessing the quality of communities, such definitions are heavily correlated. Nonetheless, the results obtained for *TPR*, *CutRatio* and *Density* (i.e. the representative metrics for each group) did not show statistical differences amongst the results for the different combinations of alternatives. Consequently, results are only reported for *FlakeODF*. On the other hand, no correlation was found amongst *ContentCohesiveness* and *Entropy*. Similar results were observed for the *Flickr* dataset.

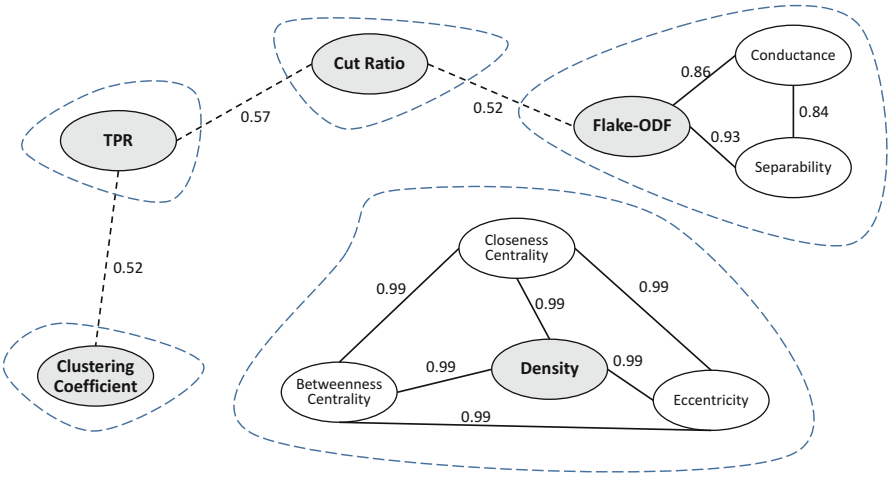


Fig. 12.1 Statistical correlation between quality assessment metrics

12.5.2 Evaluation of Community Detection Algorithms

This section presents the evaluation of the selected community detection algorithms for the *Twitter* (Sect. 12.5.2.1) and *Flickr* (Sect. 12.5.2.2) datasets. For both datasets, a statistical analysis was performed to determine whether the differences amongst results were statistically significant. As data was shown not to be normal, the Friedman test for related samples was applied to the results obtained for each community detection algorithm and combination of relations. Particularly, the results obtained for a determined node relationship across all community detection algorithms were regarded as a sample. To perform the test, two hypotheses were defined: the null and the alternative hypothesis. The null hypothesis stated that no difference existed amongst the results of the different samples, i.e. the discovered communities are independent from the algorithm used for discovering them, and the observed differences are due to chance. On the contrary, the alternative hypothesis stated that the observed differences amongst the different community structures are incidental, and not due to chance.

12.5.2.1 Results for the Twitter Dataset

The reported results include all the community detection algorithms previously described with the exception of Edge Betweenness as for each evaluated graph structure this algorithm required more than 2 h of execution, thereby being only suitable for small networks, and not for real-time processing applications. Additionally, the experimental evaluation showed that the algorithms that best scaled as the network size increased were Louvain and X-Means. On the other hand, Infomap, Walktrap

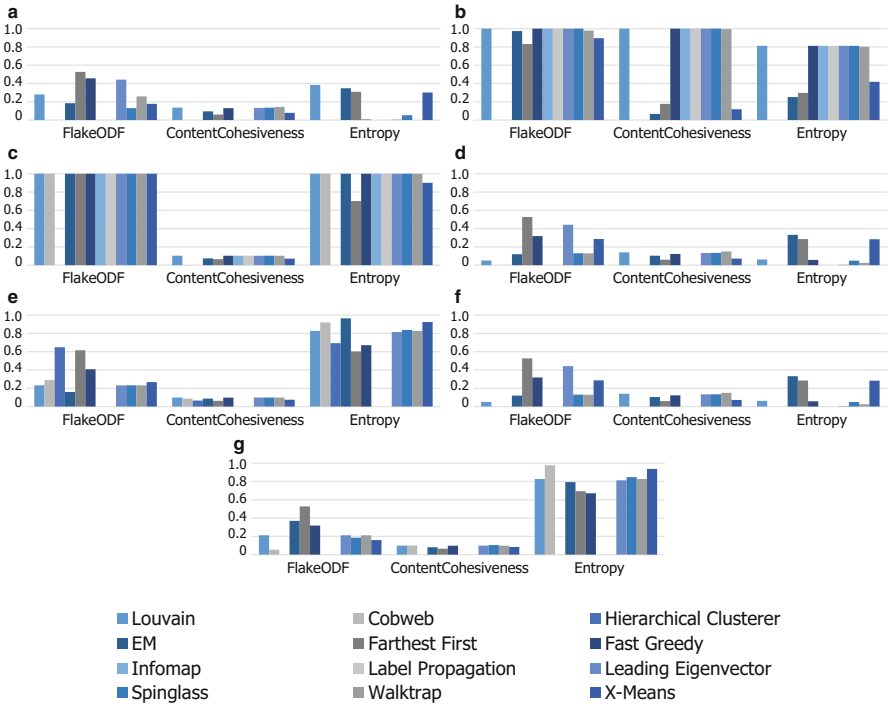


Fig. 12.2 Community detection results for the independent social and content views for the *Twitter* dataset. (**a**) *Social*. (**b**) *SimilarContent-0.6*. (**c**) *SharedClass*. (**d**) *Social* & *SimilarContent-0.6*. (**e**) *Social* & *SharedClass*. (**f**) *Social* & *SharedClass* & *SharedTag* & *SimilarContent-0.6*. (**g**) *Social* & *SharedClass* & *SharedTag* & *SimilarContent*

and Spinglass did not scale well, hindering their applicability on large networks or real-time applications. These findings are in agreement with those in [44].

Independent Social and Content Views Figure 12.2 shows the obtained results for the different combinations of node relationships for the independent derivation of the social graph. As it can be observed, there was no clear dominance of neither the traditional clustering techniques nor the algorithms specifically designed for community detection, as the quality of the obtained communities for each algorithm varied according to the node relationships under consideration.

The highest quality communities were found for high density networks. Moreover, most algorithms did not obtain stable results across the evaluated relationships. For example, Farthest First was amongst the best performing algorithms for *Social*, but also was the worst performing one for *SharedClass*. Similarly, Spinglass was one of the best performing algorithms for *SharedClass*, but the worst one for *Social*. These results show the instability and lack of robustness of algorithms when varying the underlying graph structure. Moreover, results exposed the sensitiveness (as shown in Table 12.4) of algorithms to the network degree, as described in [28]. For

example, the quality of the communities found by Spinglass and X-Means increased as the average degree of the network increased. On the other hand, for Leading Eigenvector the effect was the inverse, i.e. the quality of the found communities decreased as the average node degree increased. Finally, Walktrap consistently obtained communities of lower quality than other algorithms, independently of network density. Label Propagation only obtained results for the combinations of relationship with the lowest average node degree.

Interestingly, in the case of *Social*, several community detection algorithms found community partitions with higher *ContentCohesiveness* than when considering *SharedClass*, even though explicit content information was not included. Moreover, those same algorithms obtained the highest *ContentCohesivess* for *SimilarContent-0.6*, with competitive *Entropy* and *FlakeODF* results. These differences were maintained when both relationships were combined. The obtained results might also expose the redundancy of relationships, as combining either two of four relationships obtained similar results for most algorithms. Particularly, results were similar for *Social* & *SharedClass* (combination of two relationships) and *Social* & *SharedClass* & *SharedTag* & *SimilarContent-0.6* (combination of four relationships). In those cases, all algorithms discovered partitions with low *Content-Cohesiveness* even though, the individual relationships allowed discovering content cohesive communities.

As regards the differences between the clustering and the community detection algorithms, a curious phenomenon appeared when analysing the results for *Social*, *SharedClass* and their combination. Whilst for *Social*, results did not show a clear dominance of any type of technique, for *SharedClass*, the best average results were obtained by the community detection techniques. However, when combining both relations the tendencies were reverted, and clustering techniques obtained the best results. This phenomenon emphasises the instability and sensitivity of algorithms.

Finally, it is worth noting that the Louvain algorithm was able to find relatively high-quality communities across almost every analysed individual and combinations of relationships, regardless of the density of the analysed graph. Particularly, the performed Wilcoxon test showed with a confidence of 0.01 the statistical superiority of Louvain regarding EM and Spinglass for the three quality metrics. When solely considering *FlakeODF* and *Entropy*, Louvain obtained statistically superior results than EM, Spinglass, Walktrap, Hierarchical Clusterer, Fast Greedy, Label Propagation, X-Means and Leading Eigenvector. These results showed the capabilities of the algorithm and its stability. Moreover, the results agree with those in [44], which stated the superiority of the communities found by the Louvain algorithm, especially for large graphs.

Weighted Social View Figure 12.3 shows the obtained results of the weighted derivation of the social graph for the different combinations of node relationships. Similarly to the results obtained for the independent graph derivation, there is no clear dominance of any type of technique, excepting for those relations considering *SimilarContent-0.6*, for which clustering algorithms consistently obtained the worst results. Interestingly, for this graph derivation, Label Propagation was able to

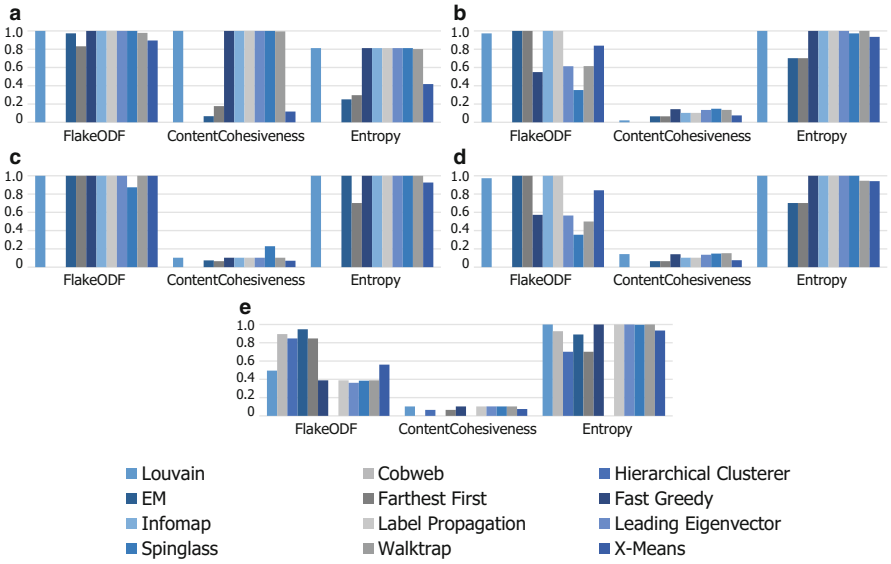


Fig. 12.3 Community detection results for the weighted social views for the *Twitter* dataset. (**a**) *Social-W-SimilarContent-0.6*. (**b**) *Social-W-SharedClass*. (**c**) *Social-W-SimilarContent-0.6* & *SharedClass*. (**d**) *Social-W-SharedClass* & *SimilarContent-0.6*. (**e**) *Social-W-SharedTag* & *SharedClass*

find a meaningful number of communities for every analysed relation. Its results were competitive regarding other techniques in all cases, but for the relationship yielding the highest node average degree. These results showed the sensitivity of the algorithm to the network degree distribution. As regard the diverse evaluated relations, results evidenced that techniques were dominated by the information provided by *SharedClass*, i.e. algorithms obtained almost equal results in every combination of relationships including *SharedClass*.

EM and Fast Greedy were shown to have contrasting results. Whilst EM was amongst the best performing techniques for *Social-W-SharedTag* & *SharedClass* and the worst one for *Social-W-SimilarContent-0.6*, Fast Greedy obtained exactly the reverse results. These results continued to expose the sensitivity of techniques towards the underlying graph distribution. Additionally, Farthest First was shown to improve the quality of the found communities as the density of the graph increased. Walktrap continued to exhibit poor performance for this graph derivation. Unlike for the independent graph derivation, the quality of the communities found by Spinglass decreased as the node average degree increased. The results for the Louvain algorithm continued to be the most stable ones across all relationships, confirming the robustness of the algorithm for finding high-quality communities. Finally, similarly as for the other graph derivation, a Wilcoxon test confirmed with a confidence of 0.01 the superiority of Louvain regarding other evaluated algorithms.

12.5.2.2 Results for the Flickr Dataset

Similarly as for the *Twitter* dataset, no results are reported for Edge Betweenness due to the excessive execution time. Additionally, no results are reported for Coweb as it did not allow to obtain a significant number of communities for any of the selected combinations of relationships. As for the *Twitter* dataset, the algorithms that best scaled as the network size increased were Louvain and X-Means. Moreover, unlike for the *Twitter* dataset, Label Propagation obtained a meaningful number of communities for every analysed combinations of relationships.

Independent Social and Content Views Figure 12.4 shows the obtained results for the different combinations of node relationships for the independent derivation of the social graph. As it can be observed, there was no clear dominance of neither the traditional clustering techniques nor the algorithms specifically designed for community detection, as the quality of the obtained communities for each algorithm varied according to the node relationships under consideration. The highest differences were observed for *ContentCohesiveness*, which achieved the lowest results for the *Social* relationship.

Similarly as for the *Twitter* dataset, the best results were found for high density networks. Nonetheless, none relationship or combination of them was able to obtain high results for the three metrics simultaneously. For example, *FlakeODF* and *Entropy* were high for *Social*, *TaggedSameUser* and *Social* & *TaggedSameUser*,

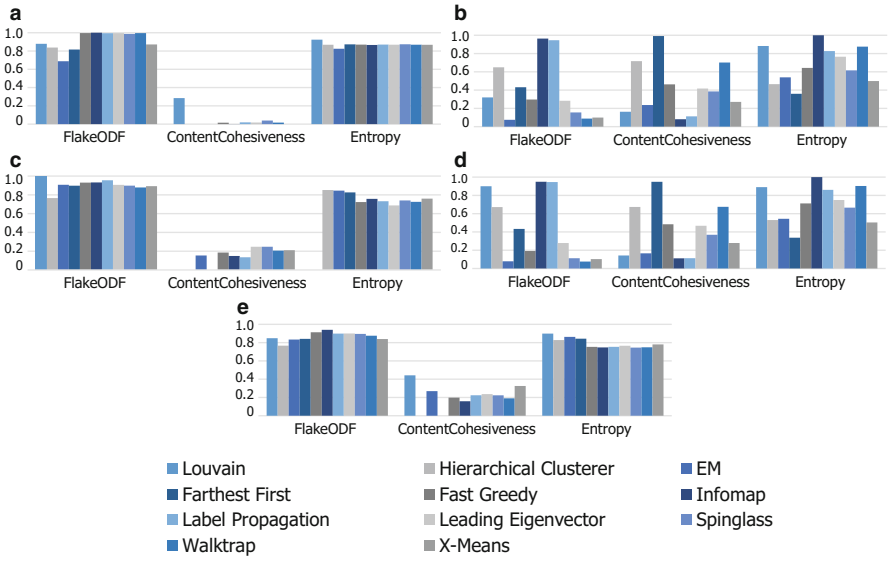


Fig. 12.4 Community detection results for the independent social and content views for the *Flickr* dataset. (**a**) *Social*. (**b**) *SimilarContent-0.6*. (**c**) *TaggedSameUser*. (**d**) *Social* & *SimilarContent-0.6*. (**e**) *Social* & *TaggedSameUser*

whilst *ContentCohesiveness* was high for *Social & SimilarContent-0.6*. Moreover, most algorithms did not obtain stable results across the evaluated relationships. For example, EM was amongst the best performing algorithms for *TaggedSameUser* and *Social & TaggedSameUser*, but also was the worst performing one for the other three combinations of relationships. Similarly, Infomap was the best performing algorithm for *SimilarContent-0.6* and *Social & SimilarContent-0.6*, but one of the worst for *Social & TaggedSameUser*. These results show the instability and lack of robustness of algorithms when varying the underlying graph structure. The sensitiveness of algorithms to network degree observed for the *Twitter* dataset was also observed for this dataset (as shown in Table 12.5); for example, the relative quality of the communities found by X-Means and EM increased as the average degree of the network increased. Interestingly, the quality of Spinglass results changed independently of the underlying node degree.

The obtained results might also expose the redundancy of relationships, as combining two relationships obtained similar results to those obtained for the individual relationships for several algorithms. For example, results were similar for *Social* and *Social & SimilarContent-0.6*, and for *TaggedSameUser* and *Social & TaggedSameUser*. These results might imply that the characteristics of the discovered communities are dominated by only one relationship of the pair.

As regards the differences between the clustering and the community detection algorithms, results did not show a clear dominance of any type of technique, as both community detection and clustering techniques achieved both high- and low-quality results. Nonetheless, in all cases the best results were obtained by community detection techniques (Infomap and Louvain), followed, in some cases, by clustering techniques, whilst the worst results were obtained by clustering techniques (EM and Hierarchical Clusterer). This phenomenon emphasises the instability and sensitivity of algorithms. Finally, it is worth noting that the Louvain algorithm obtained the best average results for four of the analysed individual and combinations of relationships, regardless of the density of the analysed graph. These results continue to show the capabilities of the algorithm and its robustness.

Weighted Social View Figure 12.5 shows the results obtained for the weighted derivation of the social graph for the different combinations of node relationships. Unlike the results observed for the independent graph derivation, the worst quality communities were consistently obtained by two clustering techniques X-Means and EM. On the other hand, the best results were obtained by community detection techniques in all cases.

For this graph derivation, the differences of average node degree were lower than for the independent derivation, excepting for *Social-W-SimilarContent & SimilarContent-0.6* (the relationship combination showing the highest average node degree), whose average degree was a 98% higher than the second highest one (*Social-W-SharedClass & SimilarContent-0.6*). In this regard, most of the techniques showed stable results for four of the five combinations of relationships analysed. Then, in the case of EM, Spinglass and X-Means, the quality of the

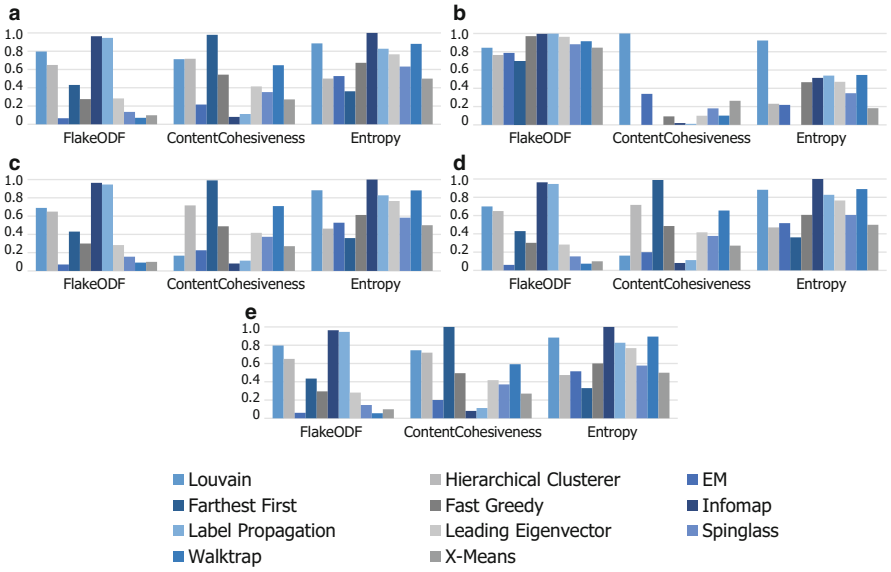


Fig. 12.5 Community detection results for the weighted social views for the *Flickr* dataset. (**a**) *Social-W-SharedClass & SimilarContent-0.6*. (**b**) *Social-W-SimilarContent & SimilarContent-0.6*. (**c**) *Social-W-TaggedSameUser & SharedClass*. (**d**) *Social-W-CommentedSameUser & SimilarContent-0.6*. (**e**) *Social-W-SharedTag & SimilarContent-0.6*

discovered communities increased when the average node degree increased, whilst the quality of Hierarchical Clusterer, Farthest First, Infomap and Label Propagation decreased.

The results for Louvain, Fast Greedy and Leading Eigenvector were the most stable ones across all relationships. These results confirm the robustness of Louvain for community detection on heterogeneous graph structures. Finally, similarly as for the independent graph derivation, a Wilcoxon test confirmed with a confidence of 0.05 the superiority of Louvain regarding other evaluated algorithms.

12.5.3 Evaluation of Vertex Similarity Metrics

This section presents the evaluation of the selected vertex similarity metrics for the *Twitter* (Sect. 12.5.3.1) and *Flickr* (Sect. 12.5.3.2) datasets. Similarly to the evaluation of the community detection techniques, the stability of the vertex similarity metrics across the different evaluated graph sizes was statistically analysed. As data was shown not to be normal, the Kruskal–Wallis test for unrelated samples was applied to the results obtained for each metric and each of the analysed combinations of relations. The results obtained for each graph size were regarded as a sample. The confidence value was set to 0.01. Results showed that the hypothesis that no

difference existed amongst the results of the different samples could not be rejected, as either the p-value was higher than the set confidence value or the critical value was higher than the obtained statistic value. Hence, the diverse vertex similarity metrics could be assumed to behave stable across the different graph sizes. For clarity of presentation, the results across different graph sizes are summarised by their mean value.

Finally, a statistical analysis was performed to determine whether the differences amongst results were statistically significant. As data was shown not to be normal, the Friedman test for related samples was applied to the results obtained for each vertex similarity metric and combination of relations. Particularly, the results obtained for a determined node relationship across all vertex similarities were regarded as a sample. The null hypothesis stated that no difference existed amongst the results of the different samples, i.e. discovering communities based on the full node set yielded the same quality than iteratively extracting one node from the set and inserting it using vertex similarity metrics. On the contrary, the alternative hypothesis stated that inserting a node in an already computed community structure does not lead to the same community quality than using the full node set.

12.5.3.1 Results for the Twitter Dataset

As the following subsections show, no differences were observed between the results for the different graph derivations. This evaluation continued to expose the redundancy amongst relationships.

Independent Social and Content Views Figure 12.6 shows the obtained results for the different combinations of node relationships for the independent derivation of the social graph. As it can be observed, results obtained for each vertex similarity metric are similar to those obtained when considering all nodes in the community detection process (named *Full Communities*). Interestingly, differences are only observable after the fourth decimal place.

The biggest differences were found for *SharedClass* (Fig. 12.6c) and *SimilarContent-0.6* (Fig. 12.6b). When considering *SharedClass*, the communities discovered for the full set of nodes had higher *ContentCohesiveness* and *Entropy* than those obtained for the vertex similarity metrics. Conversely, in the case of *SimilarContent-0.6*, the *ContentCohesiveness* obtained for the full set of nodes was lower than that observed for the vertex similarity metrics. Note that the highest differences were observed for the lowest density graph. These results could imply that community structures are very sensitive to small changes in the node set. For example, it could occur that removing one node could alter the strength of links forcing the separation of a community into two or more communities. Then, when the node is inserted in the community structure, communities are not merged, altering their quality.

The results of *Social* & *SharedClass* (Fig. 12.6e), *Social* & *SharedClass* & *SharedTag* & *SimilarContent-0.6* (Fig. 12.6d) and *Social* & *SharedClass* & *SharedTag* & *SimilarContent* (Fig. 12.6g) are identical. These results reinforce the fact

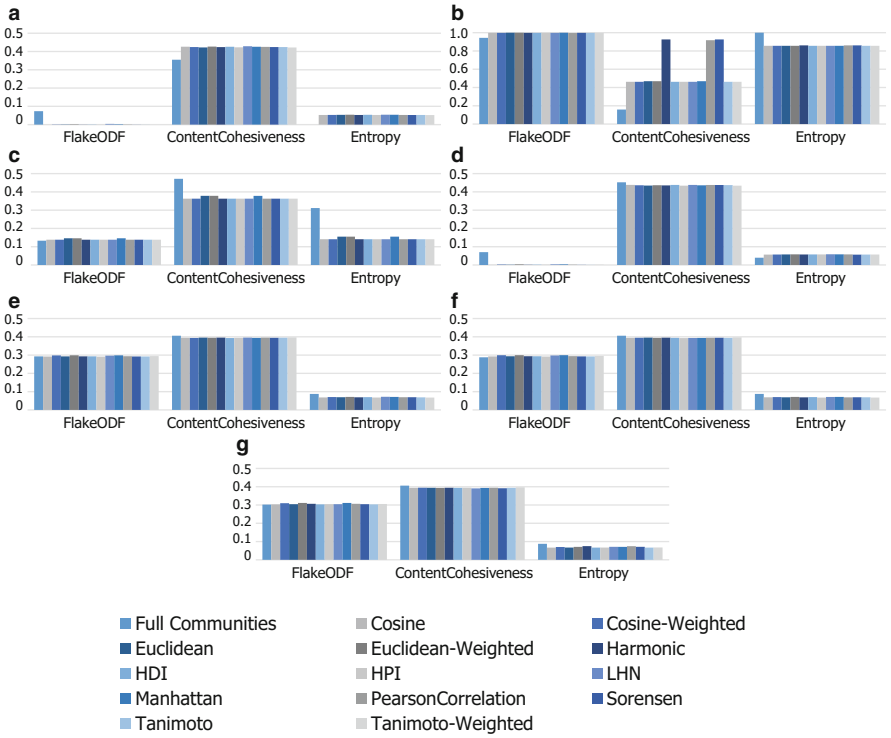


Fig. 12.6 Vertex similarity results for the independent social and content views for the *Twitter* dataset. (**a**) *Social*. (**b**) *SimilarContent-0.6*. (**c**) *SharedClass*. (**d**) *Social* & *SimilarContent-0.6*. (**e**) *Social* & *SharedClass*. (**f**) *Social* & *SharedClass* & *SharedTag* & *SimilarContent-0.6*. (**g**) *Social* & *SharedClass* & *SharedTag* & *SimilarContent*

that in some cases, adding more information does not imply an improvement of results. Instead, information sources might be redundant. As observed, for low-density graphs, *Sørensen*, *Pearson Correlation* and their combination achieved higher results than the other metrics.

Table 12.6 summarises the results obtained for each vertex similarity metric averaged for every node relationship analysed. The best three results obtained for each metric are highlighted in bold. Interestingly, in average, considering the full node set achieved the lowest *ContentCohesiveness*, but the highest *Entropy* and *FlakeODF*. Note that *Harmonic* obtained high results for the three evaluated metrics, showing the highest *ContentCohesivenness* values.

The performed Friedman test showed with a confidence of 1.229e−07 that the null hypothesis should be rejected, meaning that there is a difference between any of the analysed pair of results. To discover the pairs for which a statistical difference existed, the Wilcoxon test was applied defining the same hypotheses. Wilcoxon results showed with a confidence of 0.05 the existence of significant differences amongst the diverse vertex similarity metrics. For example, *Cosine Similarity*

Table 12.6 Summary of vertex similarity results for the independent graph derivation of the *Twitter* dataset

	FlakeODF	*ContentCohesiveness*	*Entropy*
Full Communities	**0.30**	0.38	**0.23**
Cosine	0.29	0.41	0.19
Cosine-Weighted	0.29	0.41	0.19
Euclidean	0.29	0.41	0.19
Euclidean-Weighted	0.29	0.41	0.19
Harmonic	**0.31**	**0.48**	**0.20**
HDI	0.29	0.41	0.19
HPI	0.29	0.41	0.19
LHN	0.29	0.41	0.19
Manhattan-Weighted	0.29	0.41	0.19
PearsonCorrelation	0.30	**0.48**	**0.20**
Sorensen	**0.31**	**0.48**	0.19
Tanimoto	0.29	0.41	0.19
Tanimoto-Weighted	0.29	0.41	0.19

results were shown to be statistically different and lower than most of those of the other metrics. On the other hand, $Pearson\,Correlation$ and $Harmonic$ were shown to be statistically superior to the other metrics. Finally, no statistical difference was found amongst the binary and weighted variations of the metrics.

Weighted Social View Figure 12.7 shows the obtained results for the different combinations of node relationships for the weighted derivation of the social graph. As it can be observed, the behaviour of the vertex similarity metrics did not change regarding the other graph derivation strategy. For three combinations of relationships, considering the full node set allowed to obtain higher *Entropy*, and a slight improvement of *ContentCohesiveness*. These results also exposed the redundancy amongst node relationships as *Social-W-SharedClass* (Fig. 12.7b), *Social-W-SimilarContent-0.6* & *SharedClass* (Fig. 12.7c), *Social-W-SharedClass* & *SimilarContent-0.6* (Fig. 12.9b) and *Social-W-SharedTag* & *SharedClass* (Fig. 12.7e) achieved similar results.

Table 12.7 summarises the results obtained for each vertex similarity metric averaged for every node relationship analysed. The best results obtained for each metric are in bold. Interestingly, in average, considering the full node set achieved the lowest *ContentCohesiveness* but the highest *Entropy*. Unlike for the other graph derivation, using the full node set did not discover the most structurally cohesive communities. Similarly to the previous case, $Harmonic$ was amongst the best performing vertex similarity metrics.

The same statistical analyses performed for the independent derivation results were performed for this graph derivation. The Friedman test showed with a confidence of $1.049e{-}11$ that the null hypothesis should be rejected, meaning that there was a difference between any of the analysed pair of results. Then, the

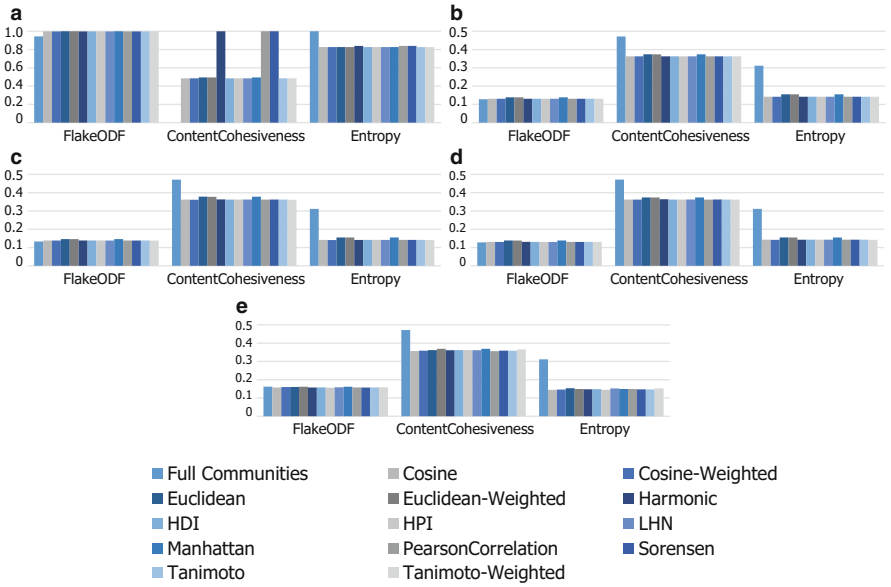


Fig. 12.7 Vertex similarity results for the weighted social views for the *Twitter* dataset. (**a**) *Social-W-SimilarContent-0.6*. (**b**) *Social-W-SharedClass*. (**c**) *Social-W-SimilarContent-0.6* & *Shared-Class*. (**d**) *Social-W-SharedClass* & *SimilarContent-0.6*. (**e**) *Social-W-SharedTag* & *SharedClass*

Table 12.7 Summary of vertex similarity results for the weighted graph derivation of the *Twitter* dataset

	FlakeODF	*ContentCohesiveness*	*Entropy*
Full Communities	0.30	0.38	**0.45**
Cosine	0.31	0.39	0.28
Cosine-Weighted	0.31	0.39	0.28
Euclidean	**0.32**	**0.40**	0.29
Euclidean-Weighted	**0.32**	**0.40**	0.29
Harmonic	**0.33**	**0.49**	**0.30**
HDI	0.31	0.39	0.28
HPI	0.31	0.39	0.28
LHN	0.31	0.39	0.28
Manhattan-Weighted	**0.32**	**0.40**	0.29
PearsonCorrelation	0.31	**0.49**	**0.30**
Sorensen	0.31	**0.49**	**0.30**
Tanimoto	0.31	0.39	0.28
Tanimoto-Weighted	0.31	0.39	0.28

Wilcoxon test was applied. Results showed with a confidence of 0.05 the existence of significant differences amongst the diverse vertex similarity metrics. For example, *Cosine Similarity* results were shown to be statistically different and lower than most of those of the other metrics, whilst *Pearson Correlation* and *Harmonic* were shown to be statistically superior to the other metrics.

12.5.3.2 Results for the Flickr Dataset

As the following subsections show, the results for this dataset present some differences regarding those observed for the *Twitter* dataset. Such differences could be due to the intrinsic and particular characteristics of each social network under analysis. Additionally, some differences were observed between the analysed graph derivations.

Independent Social and Content Views Figure 12.8 shows the obtained results for the different combinations of node relationships for the independent derivation of the social graph. As it can be observed, in terms of *FlakeODF* results obtained for each vertex similarity metric are similar to those obtained for *Full Communities*. In most cases, differences are only observed after the third decimal place. Discov-

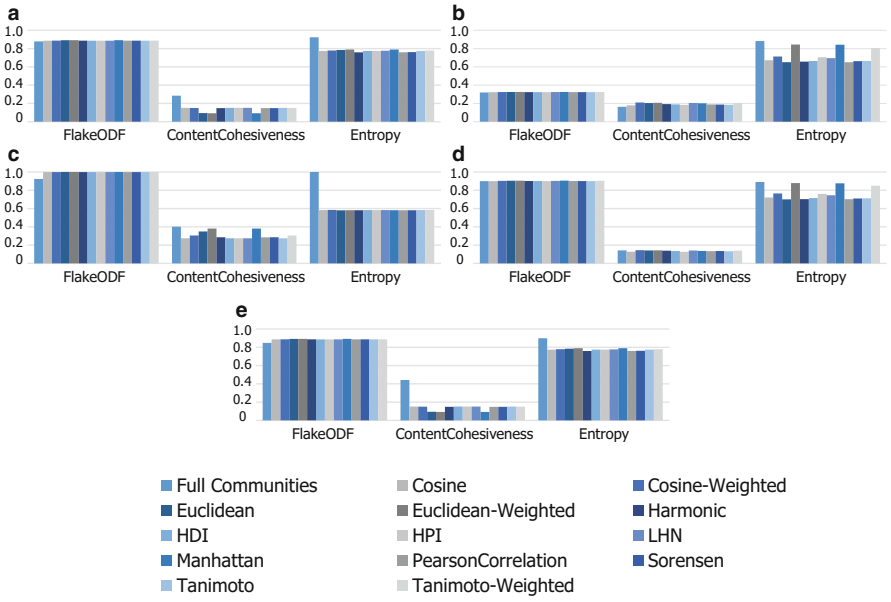


Fig. 12.8 Vertex similarity results for the independent social and content views for the *Flickr* dataset. (**a**) *Social*. (**b**) *SimilarContent-0.6*. (**c**) *TaggedSameUser*. (**d**) *Social & SimilarContent-0.6*. (**e**) *Social & TaggedSameUser*

ering communities for the full node set obtained the best average results for every combination of relationships, excepting *SimilarContent-0.6*.

In terms of *Entropy*, the biggest differences were observed for *SimilarContent-0.6* (Fig. 12.8b) and *Social & SimilarContent-0.6* (Fig. 12.8d). In those cases, the best average results were obtained when considering *Full Communities*, followed by the results of *Euclidean-Weighted*, *Manhattan-Weighted* and *Tanimoto-Weighted*. As observed for the community detection algorithms, the results for those combinations of relationships are similar, which might imply that the characteristics of the discovered communities are dominated by only one relationship of the pair. For the remaining three combinations of relationships, *Full Communities* obtained the highest *Entropy* results, with differences up to a 71% regarding the different similarity metrics. As regards *ContentCohesiveness*, for those alternatives considering *SimilarContent-0.6*, *Full Communities* did not discover the highest quality communities.

Table 12.8 summarises the results obtained for each vertex similarity metric averaged for every node relationship analysed. The best results obtained for each metric are highlighted in bold. Note that for *FlakeODF* average results were almost equal for every analysed similarity metric. Interestingly, in average, considering *Full Communities* achieved the lowest *FlakeODF*, but the highest *ContentCohesiveness* and *Entropy*. These results differ from those observed for the *Twitter* dataset, in which *FullCommunities* achieved the lowest *ContentCohesiveness* and the highest *FlakeODF*. Moreover, unlike for the *Twitter* dataset, the weighted versions of *Cosine*, *Manhattan* and *Euclidean* obtained higher results than *Harmonic*.

Table 12.8 Summary of vertex similarity results for the independent graph derivation of the *Flickr* dataset

	FlakeODF	*ContentCohesiveness*	*Entropy*
Full Communities	0.77	**0.29**	**0.92**
Cosine	0.79	0.18	0.71
Cosine-Weighted	**0.80**	**0.19**	0.72
Euclidean	0.80	0.18	0.70
Euclidean-Weighted	**0.80**	0.18	**0.78**
Harmonic	0.80	0.18	0.69
HDI	0.79	0.18	0.70
HPI	0.79	0.18	0.72
LHN	0.80	0.18	0.72
Manhattan-Weighted	**0.80**	0.18	**0.78**
PearsonCorrelation	0.80	0.18	0.69
Sorensen	0.80	0.18	0.70
Tanimoto	0.79	0.18	0.70
Tanimoto-Weighted	0.80	**0.19**	0.76

The performed Friedman test showed with a confidence of 1.229e−07 that the null hypothesis should be rejected, meaning that there is a difference between any of the analysed pair of results. To discover the pairs for which a statistical difference existed, the Wilcoxon test was applied defining the same hypotheses. Wilcoxon results showed with a confidence value of 0.05 the existence of significant differences amongst the diverse vertex similarity metrics. For example, $Euclidean$ results were shown to be statistically different and lower than most of those of the other metrics, in terms of *FlakeODF* and *ContentCohesiveness*. On the other hand, $Tanimoto\text{-}Weighted$ was shown to be statistically different than most of the other metrics in terms of *Entropy*. Interestingly, $FullCommunities$ showed to be statistically superior than several metrics, in terms of *ContentCohesiveness* for HPI, LHN, $Manhattan-Weighted$, $Tanimoto$, $Pearson$ and $Euclidean$. No significant differences were observed in terms of *FlakeODF* and *Entropy*. Finally, even though differences were observed for the binary and weighted variations of the metrics, such differences were not statistically significant.

Weighted Social View Figure 12.9 shows the obtained results for the different combinations of node relationships for the weighted derivation of the social graph. For every combination of relationships, considering the full node set allowed to obtain the highest *Entropy* results. Moreover, for every combination

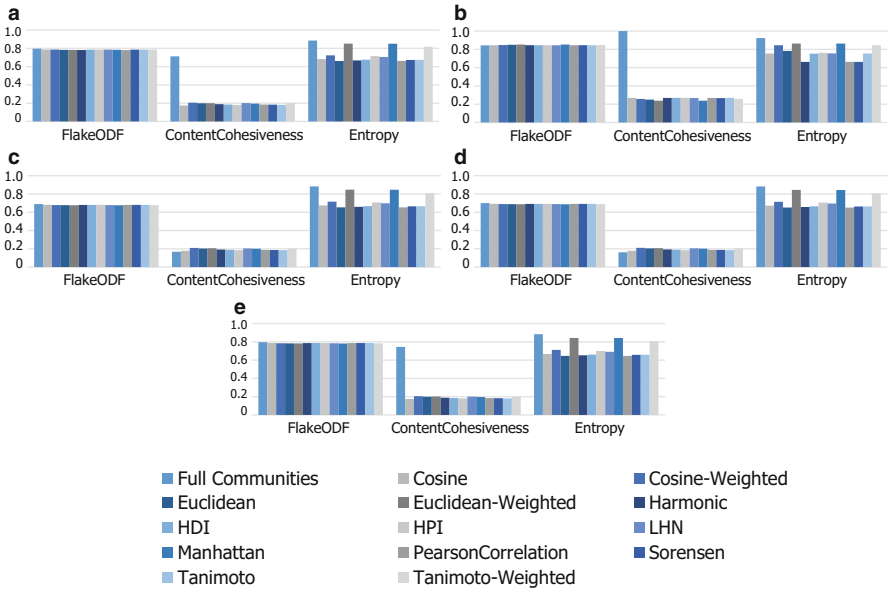


Fig. 12.9 Vertex similarity results for the weighted social views for the *Flickr* dataset. (**a**) *Social-W-SharedClass & SimilarContent-0.6*. (**b**) *Social-W-SimilarContent & SimilarContent-0.6*. (**c**) *Social-W-TaggedSameUser & SharedClass*. (**d**) *Social-W-CommentedSameUser & SimilarContent-0.6*. (**e**) *Social-W-SharedTag & SimilarContent-0.6*

Table 12.9 Summary of vertex similarity results for the weighted graph derivation of the *Flickr* dataset

	FlakeODF	*ContentCohesiveness*	*Entropy*
Full Communities	**0.77**	**0.56**	**0.89**
Cosine	0.76	0.20	0.69
Cosine-Weighted	0.76	**0.22**	0.74
Euclidean	0.76	0.21	0.68
Euclidean-Weighted	0.76	0.21	**0.85**
Harmonic	0.76	0.21	0.66
HDI	0.76	0.20	0.68
HPI	0.76	0.20	0.72
LHN	0.76	**0.22**	0.71
Manhattan-Weighted	0.76	**0.21**	**0.85**
PearsonCorrelation	0.76	0.20	0.66
Sorensen	0.76	0.20	0.66
Tanimoto	0.76	0.20	0.68
Tanimoto-Weighted	0.76	0.21	0.82

excepting *Social-W-TaggedSameUser & SimilarContent-0.6*, *FullCommunities* obtained the highest *ContentCohesiveness,* with differences up to a 300%. These results also exposed the redundancy amongst node relationships as *Social-W-SharedTag & SimilarContent-0.6* and (Fig. 12.9e) *Social-W-SharedClass & SimilarContent-0.6* (Fig. 12.9a) obtained similar results. The same applies for *Social-W-TaggedSameUser & SimilarContent-0.6* (Fig. 12.9c) and *Social-W-CommentedSameUser & SimilarContent-0.6* (Fig. 12.9d).

Table 12.9 summarises the results obtained for each vertex similarity metric averaged for every node relationship analysed. The best results obtained for each metric are in bold. Similarly as for the independent graph derivation, all metrics obtained similar average *FlakeODF*. However, unlike for the independent graph derivation, *Full Communities* obtained the highest average results. The highest differences were observed for *ContentConhesiveness*. The same statistical analyses performed for the independent derivation results were performed for this graph derivation. Results showed the same tendencies than for the independent graph derivation.

12.6 Conclusions

Social networking and microblogging sites have increased their popularity in recent years attracting millions of users, who spend an increasing amount of time on those sites sharing personal information and making new friends. For example, sites like *Flickr*, *YouTube*, *Facebook* or *Twitter* allow users to create content, publish photos, comment on content other users shared, tag content and socially connect with other

users in the form of subscriptions or friendships. Consequently, social networking sites affect how people communicate and interact with each other.

One fundamental problem in social networks is the identification of groups of elements (users, posts or other elements) when group membership is not explicitly available. A group, or community, can be defined as a set of elements that interact more frequently or are more similar to other community members than to outsiders. Community detection has proven to be valuable in diverse domains such as biology, social sciences and bibliometrics. As a result, several community detection methods have been developed based on techniques from a variety of disciplines. Given the heterogeneity of real-world networks, one question that arises is how to effectively evaluate the algorithms.

Motivated by the lack of studies analysing the problem of community detection over real-world social media networks, this chapter focused on the analysis of the performance of community detection algorithms over such type of networks (particularly over *Twitter* and *Flickr*) including the effect of diverse metrics for assessing community membership. To that end, it was also explored how to quantify the structural properties of the discovered communities in terms of several quality metrics. The obtained results exposed the sensitivity of community detection algorithms to the density and structure of the underlying graph distribution, and hence their lack of robustness. Results showed that the Louvain algorithm achieved high-quality communities for almost every analysed combination of relationships, reinforcing its capabilities and stability for accurately discovering community structures, as claimed by [44]. Moreover, the study showed the relation and dependence of several quality metrics. Finally, as regards community membership, most of the analysed metrics obtained similar results. Nonetheless, those results varied according to the analysed dataset, highlighting the importance of considering the intrinsic characteristics of the social network under analysis for the community detection process.

References

1. Arenas, A., Díaz-Guilera, A., Pérez-Vicente, C.J.: Synchronization reveals topological scales in complex networks. Phys. Rev. Lett. **96**, 114102 (2006)
2. Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech: Theory Exp. **2008**(10), P10008 (2008)
3. Brandes, U., Gaertler, M., Wagner, D.: Experiments on graph clustering algorithms. In: Di Battista, G., Zwick, U. (eds.) Algorithms – ESA 2003, pp. 568–579. Springer, Berlin/Heidelberg (2003)
4. 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, CIVR '09, pp. 48:1–48:9. Association for Computing Machinery, New York (2009)
5. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Phys. Rev. E **70**, 066111 (2004)

6. Condon, A., Karp, R.M.: Algorithms for graph partitioning on the planted partition model. Random Struct. Algoritm. **18**(2), 116–140
7. Corder, G.W., Foreman, D.I.: Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach. Wiley, Hoboken (2009)
8. Cui, P., Zhu, W., Chua, T.-S., Jain, R.: Social-sensed multimedia computing. IEEE MultiMedia **23**(1), 92–96 (2016)
9. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B **39**, 1–38 (1977)
10. Deza, M.M., Deza, E.: Encyclopedia of Distances. Springer, Berlin/Heidelberg (2009)
11. Fisher, D.H.: Knowledge acquisition via incremental conceptual clustering. Mach. Learn. **2**(2), 139–172 (1987)
12. Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3–5), 75–174 (2010)
13. Fortunato, S., Barthélemy, M.: Resolution limit in community detection. Proc. Natl. Acad. Sci. **104**(1), 36 (2007)
14. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proc. Natl. Acad. Sci. **99**(12), 7821–7826 (2002)
15. Gregory, S.: Fuzzy overlapping communities in networks. J. Stat. Mech. Theory Exp. **2011**(02), P02017 (2011)
16. Hochbaum, D.S., Shmoys, D.B.: A best possible heuristic for the k-center problem. Math. Oper. Res. **10**(2), 180–184 (1985)
17. Huang, A.: Similarity measures for text document clustering. In: Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC2008), Christchurch, pp. 49–56 (2008)
18. Lancichinetti, A., Fortunato, S.: Community detection algorithms: a comparative analysis. Phys. Rev. E **80**(5), 056117 (2009)
19. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Phys. Rev. E **78**, 046110 (2008)
20. Leskovec, J., Lang, K.J., Mahoney, M.: Empirical comparison of algorithms for network community detection. In: Proceedings of the 19th WWW, pp. 631–640. Association for Computing Machinery, New York (2010)
21. Lusseau, D.: Evidence for social role in a dolphin social network. Evol. Ecol. **21**(3), 357–366 (2007)
22. Malliaros, F.D., Vazirgiannis, M.: Clustering and community detection in directed networks: a survey. Phys. Rep. **533**(4), 95–142 (2013). Clustering and Community Detection in Directed Networks: A Survey.
23. Marin, A., Wellman, B.: Social network analysis: an introduction. In: Carrington, P., Scott, J. (eds.) Handbook of Social Network Analysis. Sage, London (2010)
24. McAuley, J., Leskovec, J.: Image Labeling on a Network: Using Social-Network Metadata for Image Classification, pp. 828–841. Springer, Berlin/Heidelberg (2012)
25. McDaid, A., Hurley, N.J.: Detecting highly overlapping communities with model-based overlapping seed expansion. In: 2010 International Conference on Advances in Social Networks Analysis and Mining, pp. 112–119 (2010)
26. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**(3), 036104 (2006)
27. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**(2), 026113 (2004)
28. Orman, G.K., Labatut, V.: A Comparison of Community Detection Algorithms on Artificial Networks, pp. 242–256. Springer, Berlin/Heidelberg (2009)
29. Orman, G.K., Labatut, V., Cherifi, H.: On accuracy of community structure discovery algorithms. CoRR, abs/1112.4134 (2011)
30. Orman, G.K., Labatut, V., Cherifi, H.: Towards realistic artificial benchmark for community detection algorithms evaluation. Int. J. Web Based Communities **9**(3), 349–370 (2013)
31. Papadopoulos, S., Kompatsiaris, Y., Vakali, A., Spyridonos, P.: Community detection in social media – performance and application considerations. Data Min. Knowl. Discov. **24**(3), 515–554 (2012)

32. Pelleg, D., Moore, A.W.: X-means: extending k-means with efficient estimation of the number of clusters. In: Proceedings of the 17th ICML, pp. 727–734. Morgan Kaufmann Publishers, San Francisco (2000)
33. Pons, P., Latapy, M.: Computing Communities in Large Networks Using Random Walks, pp. 284–293. Springer, Berlin/Heidelberg (2005)
34. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E. **76**(3), 036106 (2007)
35. Reichardt, J., Bornholdt, S.: Statistical mechanics of community detection. Phys. Rev. E **74**, 016110 (2006)
36. Rosvall, M., Axelsson, D.T., Bergstrom, C.T.: The map equation. Eur. Phys. J. Spec. Top. **178**(1), 13–23 (2009). https://doi.org/10.1140/epjst/e2010-01179-1
37. Sawardecker, E.N., Sales-Pardo, M., Amaral, L.A.N.: Detection of node group membership in networks with group overlap. Eur. Phys. J. B **67**(3), 277–284 (2009)
38. Schaeffer, S.E.: Graph clustering. Comput. Sci. Rev. **1**(1), 27–64 (2007)
39. Tang, J., Wang, X., Liu, H.: Integrating Social Media Data for Community Detection. LNAI, vol. 7472, pp. 1–20 (2012)
40. Tommasel, A., Godoy, D.: Multi-view community detection with heterogeneous information from social media data. Neurocomputing **289**, 195–219 (2018)
41. Wakita, K., Tsurumi, T.: Finding community structure in mega-scale social networks. In: Proceedings of the 16th WWW, pp. 1275–1276. Association for Computing Machinery, New York (2007)
42. Xie, J., Kelley, S., Szymanski, B.K.: Overlapping community detection in networks: The state-of-the-art and comparative study. ACM Comput. Surv. **45**(4), 43:1–43:35 (2013)
43. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. In: Proceedings – IEEE International Conference on Data Mining, pp. 745–754. IEEE Computer Society (2012)
44. Yang, Z., Algesheimer, R., Tessone, C.J.: A comparative analysis of community detection algorithms on artificial networks. Sci. Rep. **6**, 30750 (2016)
45. Zachary, W.W.: An information flow model for conflict and fission in small groups. J. Anthropol. Res. **33**(4), 452–473 (1977)
46. Zubiaga, A., Spina, D., Martínez, R., Fresno, V.: Real-time classification of twitter trends. J. Assoc. Inf. Sci. Technol. **66**(3), 462–473 (2015)

Index

Deepak P, A. Jurek-Loughrey (eds.), *Linking and Mining Heterogeneous and Multi-view Data*, Unsupervised and Semi-Supervised Learning,
https://doi.org/10.1007/978-3-030-01872-6