Petra Perner (Ed.)

LNAI 10934

Machine Learning and Data Mining in Pattern Recognition

14th International Conference, MLDM 2018
New York, NY, USA, July 15–19, 2018
Proceedings, Part I

Part I

Springer

Petra Perner (Ed.)

Machine Learning and Data Mining in Pattern Recognition

14th International Conference, MLDM 2018
New York, NY, USA, July 15–19, 2018
Proceedings, Part I

Springer

Editor
Petra Perner
Institute of Computer Vision and Applied Computer Sciences
Leipzig
Germany

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Artificial Intelligence
ISBN 978-3-319-96135-4 ISBN 978-3-319-96136-1 (eBook)
https://doi.org/10.1007/978-3-319-96136-1

Library of Congress Control Number: 2018948144

LNCS Sublibrary: SL7 – Artificial Intelligence

Printed on acid-free paper

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

Preface

The 14th event of the International Conference on Machine Learning and Data Mining MLDM 2018 was held in New York (www.mldm.de) running under the umbrella of the World Congress Frontiers in Intelligent Data and Signal Analysis, DSA2017 (www.worldcongressdsa.com).

After the peer-review process, we accepted 86 high-quality papers for oral presentation. The topics range from theoretical topics for classification, clustering, association rule and pattern mining to specific data-mining methods for the different multimedia data types such as image mining, text mining, video mining, and Web mining. Extended versions of selected papers will appear in the international journal *Transactions on Machine Learning and Data Mining* (www.ibai-publishing.org/journal/mldm).

The tutorial days rounded up the high quality of the conference. Researchers and practitioners got excellent insight into the research and technology of the respective fields, the new trends, and the open research problems that we would like to study further.

A tutorial on “Data Mining,” a tutorial on “Case-Based Reasoning,” a tutorial on “Intelligent Image Interpretation and Computer Vision in Medicine, Biotechnology, Chemistry, and the Food Industry,” and a tutorial on “Standardization in Immunofluorescence” were held before the conference.

We would like to thank all reviewers for their highly professional work and their effort in reviewing the papers. We would also like to thank the members of Institute of Applied Computer Sciences, Leipzig, Germany (www.ibai-institut.de), who handled the conference as secretariat. We appreciate the help and cooperation of the editorial staff at Springer, and in particular Alfred Hofmann, who supported the publication of these proceedings in the LNAI series.

Last, but not least, we wish to thank all the speakers and participants who contributed to the success of the conference. We hope to see you in 2019 in New York at the next World Congress (www.worldcongressdsa.com) on Frontiers in Intelligent Data and Signal Analysis, DSA2018, which combines the following three events: International Conferences on Machine Learning and Data Mining, MLDM (www.mldm.de), the Industrial Conference on Data Mining, ICDM (www.data-mining-forum.de), and the International Conference on Mass Data Analysis of Signals and Images in Medicine, Biometry, Drug Discovery Biotechnology, Chemistry, and Food Industry, MDA.

July 2018 Petra Perner

Organization

Program Chair

Petra Perner	IBaI Leipzig, Germany

Program Committee

Sergey Ablameyko	Belarus State University, Belarus
Reneta Barneva	The State University of New York at Fredonia, USA
Michelangelo Ceci	University of Bari, Italy
Patrick Bouthemy	Inria VISTA, France
Xiaoqing Ding	Tsinghua University, P.R. China
Christoph F. Eick	University of Houston, USA
Ana Fred	Technical University of Lisbon, Portugal
Giorgio Giacinto	University of Cagliari, Italy
Makato Haraguchi	Hokkaido University of Sapporo, Japan
Dimitris Karras	Chalkis Institute of Technology, Greece
Adam Krzyzak	Concordia University, Canada
Thang V. Pham	University of Amsterdam, The Netherlands
Linda Shapiro	University of Washington, USA
Tamas Sziranyi	MTA-SZTAKI, Hungary
Francis E. H. Tay	National University of Singapore, Singapore
Alexander Ulanov	HP Labs, Russia
Zeev Volkovich	ORT Braude College of Engineering, Israel
Patrick Wang	Northeastern University, USA

Additional Reviewers

Mohammad Daneshzand	University of Bridgeport, UK
Yong Jin	Wuhan FiberHome Potevio IT Ltd., P.R. China
Walid Atwa	Walid Atwa, Egypt
Soheila Abrishamin	Florida State University, USA
Piyush Kumar	Florida State University, USA
Aminata Kane	Concordia University, Canada
Yunlong Wang	University of Minnesota, USA
Huan Huo	University of Shanghai for Science and Technology, P.R. China
Terrence Fries	Indiana University of Pennsylvania, USA
Carlos Escobar	General Motors/Tecnológico de Monterrey, USA
Olga Krasotkina	Tula State University, Russia
Juliane Perner	Cancer Research Cambridge, UK
Jason Wang	New Jersey Institute of Technology, USA

Turki Turki	King Abdulaziz University, Saudi Arabia
Mohamed Bouguessa	University du Quebec a Montreal, Canada
Anit Bhandari	University Hospital Schleswig-Holstein, Germany
Carolina Carvalho	Fluminense Federal University, Brazil
Rekar Mohammed	Salahaddin University-Erbil, Iraq
Francky Fouedjio	Stanford University, USA
Vahid Behzadan	Kansas State University, USA
Evgeny Burnaev	Skolkovo Institute of Science and Technology, Russia
Alexander Bernstein	Skolkovo Institute of Science and Technology, Russia
Dymitr Ruta	Bournemouth University, UK
Dan Simovici	University of Massachussetts Boston, USA
Zeev Volkovich	ORT Braude College of Engineering, Israel
Markus Vattulainen	University of Tampere, Finland
Xiaobing Liu	Oklahoma State University, USA
Lumin Zhang	Max Plank Institute for Intelligent Systems, Germany

Contents – Part I

Contents – Part II

MaxMin Linear Initialization for Fuzzy C-Means

Aybüke Öztürk[1,2](✉), Stéphane Lallich[1], Jérôme Darmont[1], and Sylvie Yona Waksman[2]

[1] ERIC EA 3083, Université de Lyon, Lyon 2, 5 avenue Pierre Mendès France, 69676 Bron Cedex, France
{aybuke.ozturk,stephane.lallich,jerome.darmont}@univ-lyon2.fr
[2] ArAr UMR 5138, Université de Lyon, Lyon 2, 7 rue Raulin, 69365 Lyon Cedex 7, France
yona.waksman@mom.fr

Abstract. Clustering is an extensive research area in data science. The aim of clustering is to discover groups and to identify interesting patterns in datasets. Crisp (hard) clustering considers that each data point belongs to one and only one cluster. However, it is inadequate as some data points may belong to several clusters, as is the case in text categorization. Thus, we need more flexible clustering. Fuzzy clustering methods, where each data point can belong to several clusters, are an interesting alternative. Yet, seeding iterative fuzzy algorithms to achieve high quality clustering is an issue. In this paper, we propose a new linear and efficient initialization algorithm *MaxMin Linear* to deal with this problem. Then, we validate our theoretical results through extensive experiments on a variety of numerical real-world and artificial datasets. We also test several validity indices, including a new validity index that we propose, *Transformed Standardized Fuzzy Difference* (TSFD).

Keywords: Clustering · Fuzzy C-Means · Seeding · Initialization Maxmin linear method · Validity indices

1 Introduction

Clustering is a useful technique for grouping a set of unlabelled data points (instances) described by attributes (variables), such that points belonging to the same cluster (group) have similar characteristics, while points in different clusters have dissimilar characteristics. There are several types of clustering schemes, such as crisp, overlapping or fuzzy partitions, and hierarchies. Crisp clustering considers that each data point belongs to one and only one cluster. Contrary to crisp clustering, fuzzy clustering [1] considers that a data point can belong to more than one cluster. There are some situations where fuzzy clustering is very useful. For instance, let us consider three clusters achieved when categorizing textual documents: an economy cluster (topic), an energy cluster, and a politics cluster. Then a document containing the keyword "petrol" could belong to all three clusters. Moreover, fuzzy clustering helps opening a discussion with domain experts regarding clustering results.

The primary objective of our paper is to avoid using highly complex clustering methods. One solution is to use iterative fuzzy methods such as Fuzzy C-Means (FCM) and Fuzzy K-Medoids. Both methods adapt the principle of the K-Means algorithm [2]. FCM, proposed by [3] and extended by [4], applies on numerical data, while Fuzzy K-Medoids [5] applies on categorical data. Since numerical data are the most common case, we choose to experiment our proposals with FCM.

The aim of the FCM algorithm is to minimize the fuzzy within-inertia FW (see Eq. 1). Fuzzy inertia FI (see Eq. 2) composes of the FW and the fuzzy between-inertia FB (see Eq. 3). FW, FI, and FB are computed from a membership matrix U, which stores the membership coefficients u_{ik} of data point i to cluster k. Note that $FI = FW + FB$. Moreover, FI is not constant because it depends on u_{ik} value. When FW changes, the values of FI and FB also change.

$$FW = \sum_{i=1}^{n} \sum_{k=1}^{K} u_{ik}^{m} d^2(x_i, c_k) \tag{1}$$

$$FI = \sum_{i=1}^{n} \sum_{k=1}^{K} u_{ik}^{m} d^2(x_i, \overline{x}) \tag{2}$$

$$FB = \sum_{i=1}^{n} \sum_{k=1}^{K} u_{ik}^{m} d^2(c_k, \overline{x}) \tag{3}$$

where n is the number of instances, K is the number of clusters, m is the fuzziness coefficient (by default, $m = 2$. If $m = 1$, clustering is crisp. If $m > 1$, clustering becomes fuzzy), c_k is the center of the k^{th} cluster $\forall$ k, $1 \leq k \leq K$, $\overline{x}$ is the grand mean (the arithmetic mean of all data, see Eq. 4), and function $d^2()$ computes the squared Euclidean distance.

$$\overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{4}$$

FCM starts by choosing K data points as initial centroids of the clusters. Then, membership matrix values u_{ik} (see Eq. 5) are assigned to each data point in the dataset. Centroids of clusters c_k are updated based on Eq. 6 until a termination criterion is reached successfully. In FCM, this criterion can be a fixed number of iterations t, e.g., $t = 100$. Alternatively, a threshold ϵ can be used, e.g., $\epsilon = 0.0001$. Then, the algorithm stops when the relative difference of objective function $< \epsilon$.

$$u_{ik} = \frac{1}{\sum_{j=1}^{K} \left(\frac{\|x_i - c_k\|^2}{\|x_i - c_j\|^2} \right)^{\frac{1}{m-1}}} \tag{5}$$

$$c_k = \frac{\sum_{i=1}^{n} (u_{ik}^{m}) x_i}{\sum_{i=1}^{n} (u_{ik}^{m})} \tag{6}$$

When using FCM, an important point is the way of choosing K data points as initial centroids (seeds). An efficient initialization method should be linear, so that the FCM algorithm stays linear, too. Then, the initialization method must be evaluated using validity indices that are well suited to the fuzzy case.

To obtain a good validated clustering result, one has to minimize intra-cluster distance (compactness) and at the same times, one has to maximize inter-cluster distance (separability). The more often, proposed clustering validity indices associate a compactness index with a separability index.

Thence, we propose in this paper (1) a linear and efficient initialization method for FCM clustering called *MaxMin Linear*. Moreover, to compare our proposal with several initialization methods from the literature, we also propose (2) a new clustering validity index called *Transformed Standardized Fuzzy Difference* (TSFD), which is tailored to the fuzzy case. We perform validation experiments on several numerical real-world and artificial datasets.

The remainder of this paper is organized as follows. Section 2 presents initialization methods for iterative clustering and several clustering validity methods proposed in the literature. Sections 3 and 4 detail our contributions, i.e., the *MaxMin Linear* initialization method and the TSFD validity index, respectively. Section 5 deals with the experimental evaluation of the *MaxMin Linear* initialization method on several datasets, using several validity indices, including TSFD. Finally, we conclude this paper and provide some perspectives in Sect. 6.

2 Related Works

Most initialization methods are studied through K-Means clustering [2] concepts. We have reviewed various works from the literature, including much-cited papers [6–8]. In our study, we make use of commonly mentioned linear methods from these three papers.

The first initialization method by [2] uses the first K data points as centroids. This method is sensitive to the order of data. It is used by default in SPSS [9]. The second method by MacQueen (*MacQueen2*) takes K random data points as centroids. Moreover, [10] proposes to perform multiple relaunches of *MacQueen2*. Among the different relaunches, the one that optimizes FW (Eq. 1) is considered the best candidate. This method is the standard way for initializing clusters. Its main disadvantage is that already selected points are not considered when a new seed is chosen. The second disadvantage is that outliers can be chosen. On the other hand, multiple runs ensure to improve the quality of the chosen sample.

Hand et al. [11], propose an extension of Faber's method that starts with a random set of seeds. It suggests iteratively modifying the partition by randomly moving some points to other clusters. The partition minimizing FW is chosen as the best candidate. To move each data point to another random cluster, a probability α, e.g., $\alpha = 0.3$, must be set. The method is only interesting if parameter α is fixed for different datasets.

Bradley and Fayyad's method [12] starts by randomly partitioning the dataset into J subsets. Then, each subset is clustered with the K-Means algorithm using *MacQueen2* initialization. *MacQueen2* produces J sets of centers, each containing K points. The centers of clusters are combined into a superset. Then, the superset is clustered by K-Means J times. Each time, K-Means is initialized with a different center set, and members of the center set that give the smallest FW are selected as final centers.

The PCA-Part method [13] uses a divisive hierarchical approach based on Principal Component Analysis (PCA) [14]. The method starts with a single cluster containing the whole dataset. Then, it iteratively divides clusters with respect to FW. Clusters are divided into two sub-clusters by using a hyperplane that is orthogonal to the principal eigenvector of the cluster covariance matrix. The division process ends after K clusters are obtained.

The K-Means++ method [15] selects centroids based on a distance probability to the nearest center. First, it randomly selects an initial center $c_1 = x$ from the data point set X. Then, $d(x)$ is denoted as the shortest euclidean distance from x to its closest center. The next center c_i is randomly selected as $c_i = x' \in X$ with probability $d(x')^2 / \sum d(x)^2$.

Finally, in the literature, there are other methods having quadratic complexity [16,17]. Among quadratic methods, *MaxMin* (also called *Maximin*) [18] is particularly interesting. *MaxMin* first calculates all the paired distances between data points. Then, it chooses two centroids from the data points, which have the greatest distance to each other. Finally, the next centroid is the data point that is the farthest from its centroid. This approach helps decrease FW, which improves the homogeneity of clusters.

To summarize, Hand and Krzanowski [11] rely on user-defined parameters that may not be easy to set. *MacQueen2*, though easy to understand and implement, uses only one random sample. Faber improves the *MacQueen2*'s random sample through relaunches. In K-Means++, the random choice is replaced by a probabilistic choice and cluster homogeneity is taken into account. However, since the probabilistic selection does not always select sufficiently the large enough distance, several probabilistic samples are required and the best centers are selected from all relaunches.

In contrast, *MaxMin* constructs only one sample by decreasing FW and is thus deterministic. Thus, we can be sure that a chosen center is the best. Yet, it can be less effective than K-Means++ in the presence of outliers.

To evaluate initialization methods, we need to use fuzzy validity indices. According to [19], there are two groups of validity indices. The first group is only based on membership values and includes the partition coefficient index V_{PC} [20] (see Eq. 7; $\frac{1}{K} \leq V_{PC} \leq 1$; to be maximized) and the Chen and Linkens index V_{CL} [21] (see Eq. 8; $0 \leq V_{CL} \leq 1$; to be maximized).

$$V_{PC} = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} u_{ik}^2 \tag{7}$$

$$V_{CL} = \frac{1}{n} \sum_{i=1}^{n} max_k(u_{ik}) - \frac{1}{c} \sum_{k=1}^{K-1} \sum_{j=k+1}^{K} \left[\frac{1}{n} \sum_{i=1}^{n} min(u_{ik}, u_{ij}) \right], \tag{8}$$

where $c = \sum_{k=1}^{K-1} k$.

V_{CL} takes in consideration both compactness (first term of V_{CL}) and separability (second term of V_{CL}). The second group of fuzzy validity indices is based on associating membership values to cluster centers and data. It includes the adaptation of the Ratio index V_{FRatio} to fuzzy clustering [22] (see Eq. 9; $0 \leq V_{FRatio} \leq +\infty$; to be maximized), the penalized version of V_{FRatio} index which is the Calinski and Harabasz index V_{FCH} [22] (see Eq. 10; $0 \leq V_{FCH} \leq +\infty$; to be maximized), the Fukuyama and Sugeno index V_{FS} [23] (see Eq. 11; $-FI \leq V_{FS} \leq FI$; to be minimized), and the Xie and Beni index V_{XB} [24,25] (see Eq. 12; $0 \leq V_{XB} \leq FI/n * min\|x_j - v_k\|^2$; to be minimized).

$$V_{FRatio} = FB/FW \tag{9}$$

$$V_{FCH} = \frac{FB/(K-1)}{FW/(n-K)} = \frac{n-K}{K-1}\frac{FB}{FW} \tag{10}$$

$$V_{FS} = FW - FB \tag{11}$$

$$V_{XB} = \frac{\sum_{k=1}^{K}\sum_{i=1}^{n} u_{ik}^{m}\|x_i - v_k\|^2}{n * min_{j,k}\|v_j - v_k\|^2} \tag{12}$$

Among all the above stated validity indices, there is no single validity index that gives the best result for any dataset. Thus, there is room for a new validity index that is specifically tailored for fuzzy validation. This is why we propose the *Transformed Standardized Fuzzy Difference* index.

3 MaxMin Linear Fuzzy Clustering Initialization Method

MaxMin's simplicity and ability to build homogeneous clusters sounds very interesting. Yet, considering all paired distance between data points makes the method quadratic with respect to the number of data points. Thus, we present in this section an enhancement of *MaxMin* that makes it linear. Before introducing our changes, we first detail how *MaxMin* works in Algorithm 1 (see Sect. 2 for *MaxMin*'s principle).

In *MaxMin Linear*, we first calculate grand mean $\overline{x}$ (see Eq. 4). Then, we choose as first centroid the data point that is nearest to $\overline{x}$. The second centroid is the data point that has the largest distance to the first centroid. Thus, complexity remains linear with respect to the number of data points. Afterwards, the choice of the remaining centroids remains the same as in *MaxMin*. *MaxMin Linear* is formalized in Algorithm 2.

As a final note, the use of *MaxMin Linear* is not limited to use with FCM on numerical data, but also with Fuzzy K-Medoids [26] for categorical data clustering. Thus, *MaxMin Linear* can also be applied with heterogeneous data to construct fuzzy clustering ensemble. This makes of *MaxMin Linear* a simple but noteworthy contribution, in our opinion.

Algorithm 1. *MaxMin*

```
Require: Set of data points $X = \{x_1, ..., x_n\}$
Require: Number of clusters $K$
  {Select the first two centroids $c_1$ and $c_2$}
  $c_1, c_2 \leftarrow \text{argmax}(d^2(x_i, x_j))\ i, j = 1, ..., n$
  $K^* \leftarrow 2$ {Number of seeds}
  {Find the remaining seeds}
  while $K^* < K$ do
    for all $x_i \neq c_{k^*}\ i = 1, ..., n,\ k^* = 1, ..., K^*$ do
      $d_m^2(x_i) \leftarrow \min(d^2(x_i, c_{k^*}))$
    end for
    $K^* \leftarrow K^* + 1$
    $c_{K^*} \leftarrow \text{argmax}(d_m^2(x_i))\ i = 1, ..., n$
  end while
  return $\{c_{k^*}\}\ k^* = 1, ..., K^*$
```

Algorithm 2. *MaxMin Linear*

```
Require: Set of data points $X = \{x_1, ..., x_n\}$
Require: Number of clusters $K$
  {Select the first two centroids $c_1$ and $c_2$}
  $\overline{x} \leftarrow \frac{1}{n} \sum_{i=1}^{n} x_i$
  for $i \leftarrow 1$ to $n$ do
    $d_m^2(x_i) \leftarrow \min(d^2(\overline{x}, x_i))$
  end for
  $c_1 \leftarrow \text{argmin}(d_m^2(x_i))\ i = 1, ..., n$
  for $i \leftarrow 1$ to $n$ do
    $d_m^2(x_i) \leftarrow \max(d^2(c_1, x_i))$
  end for
  $c_2 \leftarrow \text{argmax}(d_m^2(x_i))\ i = 1, ..., n$
  $K^* \leftarrow 2$ {Number of seeds}
  {Find the remaining seeds}
  while $K^* < K$ do
    for all $x_i \neq c_{k^*}\ i = 1, ..., n, k^* = 1, ..., K^*$ do
      $d_m^2(x_i) \leftarrow \min(d^2(x_i, c_{k^*}))$
    end for
    $K^* \leftarrow K^* + 1$
    $c_{K^*} \leftarrow \text{argmax}(d_m^2(x_i))\ i = 1, ..., n$
  end while
  return $\{c_{k^*}\}\ k^* = 1, ..., K^*$
```

4 Transformed Standardized Fuzzy Difference Validity Index

Several problems must be cleaned up to obtain a good clustering, including evaluation of the validity of the clusters and choosing the number of clusters. However, it is not an easy process. Compactness and separation level might raise problems. Firstly, if the chosen number of clusters is larger than optimal one, some clusters are broken while they could be more compact. Secondly, if the chosen number of clusters is smaller than optimal one, some clusters are merged and while they could be more separated. When it comes to addressing resolve those problems, many cluster validity indices are proposed for fuzzy clustering algorithms. The objective is to find the optimal number of clusters that can validate the best description of the data structure.

The optimal number of the cluster can be determined by considering the variation of clustering validity index. It is distinguished into two cases: The first case, if the index is not monotonic with the number of clusters, we choose the value of the number of clusters which optimizes the index. The second case, if the index is monotonic, one can prefer to use a penalized version of the index.

In building TSFD, we first consider the difference $FB - FW$, which is similar to FS except for the sign). Unfortunately, $FI = FB + FW$ is not constant and $FB - FW \in [-FI, +FI]$. To take this particularity of fuzzy clustering into account, we propose to standardize $FB - FW$ by considering *Standardized Fuzzy Difference* $SFD = (FB - FW) \div FI$ instead. $SFD \in [-1, +1]$.

Finally, to obtain an index belonging to the $[0, 1]$ interval, we linearly transform SFD as $TSFD$ (see Eq. 13; equal to FB/FI; $\in [0, 1]$; to be maximized)

$$TSFD = \frac{1 + SFD}{2} = \frac{FB}{FI} \tag{13}$$

5 Experimental Validation

In this section, we aim to compare *MaxMin Linear* to state of the art initialization methods for FCM-like clustering algorithms, i.e., *MacQueen2*, Faber's, K-Means++, and repeated K-Means++ (retaining the best result). These methods are indeed the most common linear methods and are good representatives for random, probability, and distance-based methods. Moreover, they do not require any parameterization. To achieve our comparison of initialization methods, we use the indices mentioned in Sect. 2.

5.1 Datasets

Initialization methods are compared on 15 commonly used real-life datasets from the UCI Machine Learning Repository[1] and seven artificial datasets. Their characteristics are featured in Table 1.

In the case of real-life datasets, the true number of clusters in each dataset is assimilated to the number of labels. Although using the number of labels as the number of clusters is debatable, it is acceptable if the set of descriptive variables explain the labels well. In artificial datasets, the number of clusters is known by construction.

In addition, we created new artificial datasets by introducing overlapping and noise to some of the existing artificial datasets such as E1071-3, Ruspini_original, and E1071-5 datasets (see Table 1, ID 17, 18, and 21).

[1] http://archive.ics.uci.edu/ml/.

Table 1. Dataset features

ID	Datasets	# of data points	# of variables	# of clusters	Sources
1	Wine	178	13	3	UCI
2	Iris	150	4	3	UCI
3	Seeds	210	7	3	UCI
4	Original Wisconsin Breast Cancer (WBCD)	683	9	2	UCI
5	Wisconsin Diagnostic Breast Cancer (WDBC)	569	30	2	UCI
6	BUPA Liver Disorder (BUPA)	345	6	2	UCI
7	Pima	768	8	2	UCI
8	Glass	214	9	6	UCI
9	Vehicle	846	18	4	UCI
10	Segmentation	2310	19	7	UCI
11	Parkinson	150	22	2	UCI
12	Movement Libras	360	90	15	UCI
13	Ecoli	336	7	8	UCI
14	Yeast	1484	8	10	UCI
15	WineQuality-Red	1599	11	6	UCI
16	Bensaid	49	2	3	[27]
17	E1071-3	150	3	3	[28]
18	Ruspini_original	75	2	4	[1]
19	E1071-3-overlapped	150	3	3	[28]
20	Ruspini_noised	95	2	4	[1]
21	E1071-5	250	3	5	[28]
22	E1071-5-overlapped	250	3	5	[28]

To create the dataset, new data points are introduced and each must be labeled. To obtain a dataset with overlapping, we modified the construction of the E1071 artificial datasets [28]. In the original datasets, there are three or five clusters of equal size (50). Cluster i is generated according to a Gaussian distribution $N(i; 0.3)$. To increase overlapping while retaining the same cluster size, we only change the standard deviation from 0.3 to 0.4. Then, there is no labeling problem.

Noise is introduced in each cluster by adding noisy points generated by a Gaussian variable around each label gravity center. First, for each label, we calculate the coordinates of centers, and the mean and standard deviation of each variable. With Gaussian variables, points mainly lie between "center +/− two standard deviations".

Noisy data are often generated by distributions with positive skewness. For example, in a two-dimensional dataset, for each label, we add points that are far away from the corresponding gravity center, especially on the right hand side, which generally contains the most points. Then, we draw a random number r between 0 and 1. If $r \leq 0.25$, the point is attributed to the left hand side. Otherwise, the point is attributed to the right hand side. This method helps obtain noisy data that are $^1/_4$ times smaller and $^3/_4$ times greater, respectively, than the expected value for the considered label. We apply this process to the Ruspini dataset [1].

5.2 Experimental Settings

In our experiments, we parameterize the FCM algorithm as follows: default termination criterion $\epsilon = 0.0001$ and default fuzziness coefficient value $m = 2$. We used these default settings as we are only interested in improving the initialization of FCM algorithm. All initialization methods and clustering validity indices are written in Python version 2.7.4. Repeated K-Means++ runs are performed ten times.

5.3 Experimental Results

In our experiments, we compare our method *MaxMin Linear* to all initialization methods from Sect. 2, on all datasets. We account for the following comparison criteria: number of iterations, V_{PC}, V_{CL}, FB, FW, FI, V_{FRatio}, V_{TSFD}, V_{FS}, and V_{XB}. We also rank the initialization methods with respect to all criteria.

Since presenting all results would take too much space, we only present three real-life datasets i.e., WineQuality-Red (Tables 2, 3, and 4), Glass (Tables 5, 6, and 7), and Segmentation (Tables 8, 9, and 10), as well as two of the artificial datasets we modified to introduce noise and overlapping, i.e., Ruspini_noised (Tables 11, 12, and 13), and E1071-5-overlapped (Tables 14, 15, and 16), respectively. Finally, the average ranking of initialization methods on all datasets is presented in Table 17.

From these experimental results, several observations can be drawn. In regard to the number of iterations, recall that Faber's and K-Means++ $\times 10$ methods are relaunches of two stochastic initialization methods: *MacQueen2* and K-Means++, respectively. With an average ranking of 1.68 (Table 17), *MaxMin Linear* outperforms all other methods, including single-run methods *MacQueen2* (average ranking: 1.95) and K-Means++ (average ranking: 1.95).

Regarding clustering result quality, *MaxMin Linear* obtains the best average ranking for eight of the nine experimented quality indices (Table 17). Only the FB index yields a better result for the two multiple-runs methods, while the result of *Maxmin Linear* is similar to those of *MacQueen2* and K-Means++. However, *Maxmin Linear* achieves the best trade-off between FB and FW, and thus maximizes the indices that take both FB and FW into account (V_{FRatio}, V_{TSFD}, V_{FS} and V_{XB}). The best result for *MaxMin Linear* is obtained with V_{TSFD} (average ranking of 1.86; Table 17), the new index specially tailored for fuzzy clustering that we propose.

Table 2. Experiment results on WineQuality-Red (1/2)

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW
MacQueen2	45	0.664	0.7455	110972.7	1224079.7
Faber	430	0.664	0.7455	**101440.4**	1224079.7
K-Means++	37	0.616	0.7029	101440.5	1089058.1
K-Means++ ×10	393	0.664	0.7455	**101440.4**	1224073.7
MaxMin linear	**34**	**0.665**	**0.7458**	110972.7	**1224384.8**

Table 3. Experiment results on WineQuality-Red (2/2)

Initialization method	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	1335052.363	11.0305	**0.9169**	−1113107.01	0.1621
Faber	1335052.363	11.0305	0.9148	−1113107.01	0.1621
K-Means++	1190498.537	10.7359	0.9148	−987617.57	0.2388
K-Means++ ×10	1335046.425	11.0304	0.9148	−1113101.04	0.1621
MaxMin linear	**1335357.554**	**11.0332**	**0.9169**	**−1113412.13**	**0.1611**

Table 4. Ranking of initialization methods on WineQuality-Red

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	3	2	2	4	2	2	2	2	2	2
Faber	5	2	2	2	2	2	2	5	2	2
K-Means++	2	5	5	3	5	5	5	3	5	5
K-Means++ ×10	4	4	4	**1**	4	4	4	4	4	4
MaxMin linear	**1**	**1**	**1**	5	**1**	**1**	**1**	**1**	**1**	**1**

Table 5. Experiment results on Glass (1/2)

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW
MacQueen2	**44**	0.493	0.570	452.6	**154.1**
Faber	456	0.493	0.570	452.6	**154.1**
Kmeans++	56	0.493	0.570	452.6	**154.1**
K-Means++ ×10	366	0.493	0.570	452.6	**154.1**
MaxMin linear	68	**0.555**	**0.645**	**508.3**	162.9

In conclusion, the results obtained with *MaxMin Linear* are a little better than those obtained with multiple-runs methods, but they require ten times fewer iterations. Moreover, *MaxMin Linear* is deterministic, whereas multiple-runs methods are stochastic.

Table 6. Experiment results on Glass (2/2)

Initialization method	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	606.8	2.94	0.74596	-298.5	2.358
Faber	606.8	2.94	0.74597	-298.5	2.358
Kmeans++	606.7	2.94	0.74593	-298.4	2.358
K-Means++ ×10	606.7	2.94	0.74604	-298.4	2.358
MaxMin linear	**671.2**	**3.12**	**0.75725**	**-345.4**	**0.453**

Table 7. Ranking of initialization methods on Glass

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	**1**	2	2	2	**1**	2	2	4	2	5
Faber	5	3	3	3	2	3	3	3	3	2
Kmeans++	2	5	5	5	4	5	5	5	5	4
K-Means++ ×10	4	4	4	4	3	4	4	2	4	3
MaxMin Linear	3	**1**	**1**	**1**	5	**1**	**1**	**1**	**1**	**1**

Table 8. Experiment results on Segmentation (1/2)

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW
MacQueen2	103	0.381	0.476	12384361.4	5781042.6
Faber	731	0.398	0.488	14157566.6	5680259.6
Kmeans++	146	0.381	0.476	12388277.9	5781061.6
K-Means++ ×10	930	0.399	0.490	14254025.9	**5666840.5**
MaxMin linear	**54**	**0.430**	**0.526**	**19234921.0**	6344612.7

Table 9. Experiment results on Segmentation (2/2)

Initialization method	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	18165404.0	2.14	0.6818	-6603318.7	0.363
Faber	19837826.2	2.49	0.7137	-8477307.1	0.464
Kmeans++	18169339.6	2.14	0.6818	-6607216.3	0.361
K-Means++ ×10	19920866.4	2.52	0.7136	-8587185.5	**0.341**
MaxMin Linear	**25579533.7**	**3.03**	**0.7520**	**-12890308.3**	0.656

Table 10. Ranking of initialization methods on Segmentation

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	2	4	5	5	3	5	5	5	5	3
Faber	4	3	3	3	2	3	3	2	3	4
Kmeans++	3	5	4	4	4	4	4	3	4	2
K-Means++ ×10	5	2	2	2	**1**	2	2	4	2	**1**
MaxMin linear	**1**	**1**	**1**	**1**	5	**1**	**1**	**1**	**1**	5

Table 11. Experiment results on Ruspini_noised (1/2)

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW
MacQueen2	9	0.775121	0.806518	219099.6	23421.0260
Faber	130	0.775125	0.806517	219100.8	23421.0258
Kmeans++	13	0.775122	0.806521	219101.1	23421.0258
K-Means++ ×10	105	**0.775128**	0.806518	219102.3	**23421.0256**
MaxMin linear	**7**	0.775128	**0.806523**	**219105.4**	23421.0268

Table 12. Experiment results on Ruspini_noised (2/2)

Initialization method	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	242520.7	9.3548	0.903427	−195678.6	0.063680
Faber	242521.9	9.3549	0.903427	−195679.8	0.063681
Kmeans++	242522.1	9.3549	0.903427	−195680.0	0.063676
K-Means++ ×10	242523.3	9.3549	0.903426	−195681.3	0.063681
MaxMin linear	**242526.4**	**9.3551**	**0.903429**	**−195684.4**	**0.063672**

Table 13. Ranking of initialization methods on Ruspini_noised

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	2	5	3	5	4	5	5	4	5	3
Faber	5	3	5	4	2	4	4	3	4	5
Kmeans++	3	4	2	3	3	3	3	2	3	2
K-Means++ ×10	4	**1**	4	2	**1**	2	2	5	2	4
MaxMin linear	**1**	2	**1**	**1**	5	**1**	**1**	**1**	**1**	**1**

Table 14. Experiment results on E1071-5-overlapped (1/2)

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW
MacQueen2	8	0.735646	0.762681	219.7337	48.715631
Faber	103	0.735645	0.762683	219.7358	48.715630
Kmeans++	12	0.735651	0.762685	219.7408	48.715632
K-Means++ ×10	113	0.735645	0.762683	219.7363	**48.715629**
MaxMin linear	**7**	**0.735652**	**0.762688**	**219.7445**	**48.715629**

Table 15. Experiment results on E1071-5-overlapped (2/2)

Initialization method	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	268.4494	4.5105	0.818530	−171.0181	0.11574
Faber	268.4514	4.5106	0.818535	−171.0202	**0.11569**
Kmeans++	268.4565	4.5107	0.818534	−171.0252	0.11575
K-Means++ ×10	268.4519	4.5106	0.818530	−171.0207	**0.11569**
MaxMin linear	**268.4601**	**4.5108**	**0.818537**	**−171.0288**	0.11572

Table 16. Ranking of initialization methods on E1071-5-overlapped

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	2	3	5	5	4	5	5	5	5	4
Faber	4	5	4	4	3	4	4	2	4	**1**
Kmeans++	3	2	2	2	5	2	2	3	2	5
K-Means++ ×10	5	4	3	3	**1**	3	3	4	3	2
MaxMin linear	**1**	**1**	**1**	**1**	2	**1**	**1**	**1**	**1**	3

Table 17. Average ranking of initialization methods on all datasets

Initialization method	# of iteration	V_{PC}	V_{CL}	FB	FW	FI	V_{FRatio}	V_{TSFD}	V_{FS}	V_{XB}
MacQueen2	1.95	3.36	3.55	3.86	3.41	3.41	3.41	3.04	3.41	3.55
Faber	4.45	2.73	2.82	1.73	2.73	2.73	2.73	3.27	2.73	2.91
K-Means++	1.95	3.86	3.68	3.86	3.86	3.86	3.86	3.54	3.86	3.36
K-Means++ ×10	4.41	2.68	2.55	**1.64**	2.86	2.86	2.86	3.22	2.86	2.82
MaxMin linear	**1.68**	**2.27**	**2.32**	3.82	**2.05**	**2.05**	**2.05**	**1.86**	**2.05**	**2.27**

6 Conclusion and Perspectives

In this paper, we propose a new, fast, and easy to implement initialization method for FCM called *MaxMin Linear*. *MaxMin Linear* is compared to several initialization methods from the literature. It is experimentally shown that *MaxMin Linear* outperforms existing methods on 22 datasets. Moreover, we also propose an appropriate fuzzy validity index, TSFD, to evaluate initialization methods.

In addition, *MaxMin Linear* can be applied to algorithms other than FCM, such as Fuzzy K-Modes and Fuzzy K-Medoids, which apply on categorical. In particular, *MaxMin Linear* allows decreasing the complexity of Park's Fuzzy K-Medoids implementation.

In consequence, an immediate perspective to our work is to propose a new clustering ensemble method for heterogeneous datasets composed of both numerical and categorical data.

Acknowledgments. This project is supported by the Rhône Alpes Region's ARC 5: "Cultures, Sciences, Sociétés et Médiations" through A. Öztürk's Ph.D. grant.

References

1. Ruspini, E.H.: Numerical methods for fuzzy clustering. Inf. Sci. **2**(3), 319–350 (1970)
2. 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, CA, USA, vol. 1, pp. 281–297 (1967)
3. Dunn, J.C.: A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters (1973)
4. Bezdek, J.C., Ehrlich, R., Full, W.: FCM: the fuzzy c-means clustering algorithm. Comput. Geosci. **10**(2–3), 191–203 (1984)
5. Kaufman, L., Rousseeuw, P.J.: Partitioning around medoids (program PAM). In: Finding Groups in Data: An Introduction to Cluster Analysis, pp. 68–125 (1990)
6. Steinley, D., Brusco, M.J.: Initializing k-means batch clustering: a critical evaluation of several techniques. J. Classif. **24**(1), 99–121 (2007)
7. Maitra, R., Peterson, A.D., Ghosh, A.P.: A systematic evaluation of different methods for initializing the k-means clustering algorithm. IEEE Trans. Knowl. Data Eng. 41 (2011)
8. Celebi, M.E., Kingravi, H.A., Vela, P.A.: A comparative study of efficient initialization methods for the k-means clustering algorithm. Expert Syst. Appl. **40**(1), 200–210 (2013)
9. Norušis, M.J.: IBM SPSS Statistics 19 Statistical Procedures Companion. Prentice Hall, Upper Saddle River (2012)
10. Faber, V.: Clustering and the continuous k-means algorithm. Los Alamos Sci. **22**(138144.21), 138–144 (1994)
11. Hand, D.J., Krzanowski, W.J.: Optimising k-means clustering results with standard software packages. Comput. Stat. Data Anal. **49**(4), 969–973 (2005)
12. Bradley, P.S., Fayyad, U.M.: Refining initial points for k-means clustering. In: ICML, vol. 98, pp. 91–99 (1998)
13. Su, T., Dy, J.G.: In search of deterministic methods for initializing k-means and Gaussian mixture clustering. Intell. Data Anal. **11**(4), 319–338 (2007)
14. Wold, S., Esbensen, K., Geladi, P.: Principal component analysis. Chemometr. Intell. Lab. Syst. **2**(1–3), 37–52 (1987)
15. Arthur, D., Vassilvitskii, S.: k-means++: the advantages of careful seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, pp. 1027–1035 (2007)
16. Lance, G.N., Williams, W.T.: A general theory of classificatory sorting strategies: II. Clustering systems. Comput. J. **10**(3), 271–277 (1967)
17. Astrahan, M.: Speech analysis by clustering, or the hyperphoneme method. Technical report, Department of Computer Science, Stanford University, CA (1970)
18. Gonzalez, T.F.: Clustering to minimize the maximum intercluster distance. Theoret. Comput. Sci. **38**, 293–306 (1985)
19. Wang, W., Zhang, Y.: On fuzzy cluster validity indices. Fuzzy Sets Syst. **158**(19), 2095–2117 (2007)
20. Bezdek, J.C.: Cluster validity with fuzzy sets (1973)
21. Chen, M.Y., Linkens, D.A.: Rule-base self-generation and simplification for data-driven fuzzy models. In: The 10th IEEE International Conference on Fuzzy Systems, vol. 1, pp. 424–427. IEEE (2001)
22. Caliński, T., Harabasz, J.: A dendrite method for cluster analysis. Commun. Stat.-Theory Methods **3**(1), 1–27 (1974)

23. Fukuyama, Y.: A new method of choosing the number of clusters for the fuzzy c-mean method. In: Proceedings of 5th Fuzzy Systems Symposium, pp. 247–250 (1989)
24. Xie, X.L., Beni, G.: A validity measure for fuzzy clustering. IEEE Trans. Pattern Anal. Mach. Intell. **13**(8), 841–847 (1991)
25. Pal, N.R., Bezdek, J.C.: On cluster validity for the fuzzy c-means model. IEEE Trans. Fuzzy Syst. **3**(3), 370–379 (1995)
26. Park, H.S., Jun, C.H.: A simple and fast algorithm for K-medoids clustering. Expert Syst. Appl. **36**(2), 3336–3341 (2009)
27. Bensaid, A.M., Hall, L.O., Bezdek, J.C., Clarke, L.P., Silbiger, M.L., Arrington, J.A., Murtagh, R.F.: Validity-guided (re)clustering with applications to image segmentation. IEEE Trans. Fuzzy Syst. **4**(2), 112–123 (1996)
28. Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F., Chang, C.C., Lin, C.C., Meyer, M.D.: Package e1071. Version 1.6-8 (2017)

Automatic Rail Flaw Localization and Recognition by Featureless Ultrasound Signal Analysis

Valentina Sulimova[1](✉), Alexander Zhukov[2], Olga Krasotkina[3], Vadim Mottl[1,4], and Anatoly Markov[5]

[1] Tula State University, 92 Lenin Ave., Tula 300012, Russia
{vsulimova, vmottl}@yandex.ru
[2] Sberbank of Russia, 19 Vavilova St., Moscow 117997, Russia
[3] Markov Processes International, 475 Springfield Ave, Suite 401, Summit, NJ 07901, USA
o.v.krasotkina@yandex.ru
[4] Computing Center of the Russian Academy of Sciences, Vavilov St. 40, Moscow 119333, Russia
[5] Radioavionica Corporation, Troitskiy pr. 4, building B, Saint-Petersburg 190005, Russia

Abstract. Ultrasound testing is a popular technique to find some hidden rail damages. In this paper we focus on the modern Russian railway flaw detectors, such as AVICON-14, which produce the results of ultrasound testing in the form of B-scan signals. We propose an approach simple enough to do fast automatic localization of B-scan signal segments, which could contain rail flaws. In order to recognize the selected segments as flaws of some kind or not flaws we apply SVM classifier jointly with DTW-based dissimilarity measure, specifically adapted by us to B-scan signals. To improve rail flaw localization and recognition quality we preprocess B-scan signals by applying some filter and making their convergence. Fast localization procedure jointly with CUDA implementation of B-scan segments comparison possesses the possibility to process big amounts of data. The experiments have shown that all rail flaws have been localized correctly and cross-validation ROC-score = 0.82 for the rail flaw recognition has been reached.

Keywords: Russian railways · Ultrasound testing · B-scan
Rail flaw localization · Rail flaws recognition · DTW · SVM
Featureless approach

1 Introduction

The inspection of railway infrastructure components is a very important task in railway maintenance. In particular, rail flaws are exceptionally dangerous for the operation of rail traffic [1]. Non-destructive rail inspection is the main and often the only possible way to prevent emergency situations [2, 3]. However, heightened scientists' interest to rail flaws recognition problem via non-destructive inspection is comparatively recent and the problem has not a satisfying solution yet [4], especially for Russian railways' conditions.

At the present time, two main non-destructive rail inspection methods are applied: acoustic (ultrasound) [3] and magnetodynamic [5] ones. The magnetodynamic method gives the possibility to detect rail cracks in its initial stage, but it can be used only to detect flaws near rail head surface [5]. Besides, large sizes and weight of magnetodynamic devices limit the methods' application. The main advantage of ultrasound method, in contrast to the magnetodynamic one, is the possibility to scan rails in any depth and projections. But the problem of its using consists in the probable presence of uncontrolled regions, which are in most cases due to temporary bad transducer's acoustic contact with the rail surface because of adverse weather conditions [6, 7]. In some papers the techniques for joint using of both ultrasound and magnetodynamic methods at the same time are proposed for the indicated problems' decision [8]. Sometimes visual rail flaw detection systems are proposed [9, 10], which analyze information from cameras, but it is evident that such systems can be useful only for the rail head surface defect detection.

So, as long as the most widely-spread non-destructive rail testing devices in the Russian railways are ultrasound ones [11, 12], and the problem of presence of uncontrolled regions due to temporally bad acoustic contact can be successfully algorithmically solved in accordance with the method proposed by us [13], this work exclusively focuses on the ultrasound testing devices, and more specifically, on portable double-rail track flaw detectors such as AVICON-14, presented in Fig. 1.

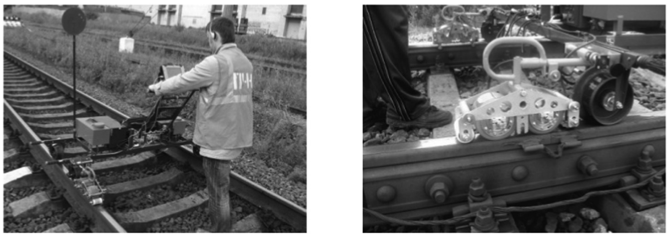


Fig. 1. The modern flaw detector AVICON-14

The operation principle of the flaw detector is based on its moving along the rail surface and emitting short ultrasound impulses. Each of these impulses, being reflected from the opposite rail surface, flaw or structural element, is registered by the detector. To have the possibility to detect differently located rail flaws of various form and orientation, a number of ultrasound channels are used, which differ by pulse angles.

Flaw detectors register signals in the form of so-called B-scans (the detailed description of B-scans is presented in the Sect. 2.1). There exists special software to register and visualize B-scan signals [14], but their analysis is done by experts manually and though to be inefficient [15].

Generally speaking, there exists a number of papers dealing with the problem of automatic analysis of ultrasonic testing results [2, 16–18 etc.]. But they can't be used for the Russian railway testing, because of the device specifics or conditions in which the testing is made.

The main problem solved in this paper is the problem of B-scan signal fragments recognition as a rail flaw of some type, or not a rail flaw. But it should be noticed that initially B-scan is a continuous signal, and so, firstly we should localize the segments, which can contain rail flaws, i.e. make B-scan signal segmentation onto segments, which can contain a rail flaw and segments without flaws.

Traditional ways for signal segmentation, which have been designed for speech [19], biomedical [20] and other signals [21, 22] have, as a rule, high computational complexity. An essentially faster approach was proposed in our previous paper [13]. It takes into account the B-scan properties that make it possible to simplify and accelerate computations. However, this approach is targeted on the localization of bolt-on rail joints and essentially exploits the assumption, that all the regions of interest have approximately equal length. In the case of rail flaw localization, the regions of interest can have different length and so this approach [13] can't be used.

In this paper we propose a fast automatic B-scan signal segmentation procedure, which takes into account B-scan signal properties and allows to make fast localization segments, which contain a rail flaw without any suppositions about the length of localized segments.

For the recognition of selected segments as flaws of some kind or not flaws we apply Support Vector Machines classifier in the featureless manner [23, 24] by B-scan segments comparison via some appropriate dissimilarity measure instead of feature vectors.

There is a number of ways to compare discrete signals [25–27]. But the most popular and traditionally used way to compare the signals of different length is based on local warping of axis of the compared signals and called the Dynamic Time Warping (DTW) [25]. In this paper we use the DTW-based approach, adapted by us for the comparing of ultrasonic B-scan signals by incorporating special dissimilarity measure of elements, which was initially proposed by us in [28] and successfully used in [13]. This allows taking into account the specificity of the considered problem.

To improve the rail flaw localization and recognition quality we preprocess B-scan signals by applying some filter and making special convergence procedure, which is described in the Sect. 2.3.

The experiments have shown that all rail flaws have been localized correctly and cross-validation ROC-score = 0.82 for the rail flaw recognition has been reached.

2 Rail Flaws Localization via Ultrasound B-Scan Signals Analysis

2.1 Representation of Ultrasound Testing Results in B-Scan Form

The most progressive way of ultrasound signals representation is so-called B-scan, which presents signals in the coordinate plane "the measured time of ultrasonic signal propagation through the rail – the coordinate of the path along the controlled rail track" [11].

In the process of ultrasound inspection several ultrasonic channels are usually used. There are ultrasonic emitters/ receivers, which send impulses of some fixed amplitude and under some angle to a rail roll surface (different channels differ by the angle of sending/receiving impulses). The signal is registered only in case the receiver obtains a signal with big enough amplitude during some time interval since the moment of sending.

Under the condition of a good enough acoustic contact and the presence of some reflective surface under 90° angle to ultrasound signal propagation, as a rule, single reflection occurres. However, when some damage or constructive reflector is located near the rail surface, the ultrasound signal has no time to fade out and multiple re-reflections can be observed. As a result a number of impulses can be registered with different delays respecting the initial moment of the emission. In case of absence of the reflection because of the bad acoustic contact or reflection by some damage turned on the angle different from 90°, the impulse will not be registered at all [11].

The presence of damages or some constructional reflectors (such as bolt-on rail joint) leads to the appearance of some lines on a B-scan. These lines can have different form, length and orientation subject to the type of an object on the way of the ultrasound signal.

In Fig. 2 (left and center) the examples of ultrasound rail B-scans with cross contact-fatigue cracks in the rail head are presented, and Fig. 2 (right) shows a picture of rail flaw of the respective kind.

2.2 Ultrasound B-Scan as Multi-component Discrete Signal

Each element of the B-scan by each channel is represented as impulse signal at the space "delay"-"amplitude" [11]. In accordance with the above description it can contain some number $n = 0, 1, 2, \ldots$ of impulses, each of which is characterized by its delay τ_i and amplitude a_i, $i = 1, \ldots, n$. For the convenience of further reasoning the element without impulses we will consider as the signal of the length $n = 1$, but with zero delay and amplitude: $\tau_1 = 0$, $a_1 = 0$. Subject to this, each element of the B-scan is represented by two-component n-length signal of pairs.

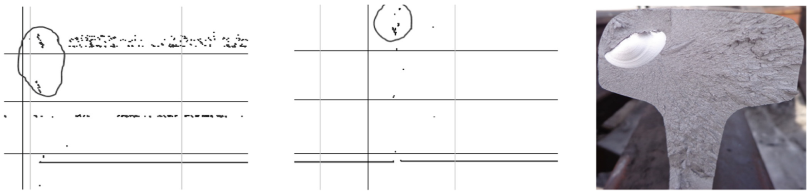


Fig. 2. Examples of ultrasound B-scans of rail head cross cracks (left and center) and an example of rail, which has the respective flaw (right)

To involve the information which is obtained from different ultrasound channels each element of B-scan is considered as m-component vector:

$$\mathbf{x} = [x_1, \ldots, x_m]^T,\ x_k = [(\tau_i,\ a_i) \in R^2,\ i = 1, \ldots, n],\ \ k = 1, \ldots, m. \tag{1}$$

So, each B-scan is m-component $N_{\mathbf{X}}$-length discrete signal $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_{N_{\mathbf{X}}})$.

2.3 B-Scan Convergence Procedure

It should be noticed that the data which is obtained by a multi-channel flaw detector has the following important feature: for any B-scan element $\mathbf{x}_s$, $s = 1, \ldots, N_{\mathbf{x}}$ the index s characterizes the flaw detector's coordinate at the moment of signals $x_{s,1}, \ldots, x_{s,m}$ registration, but each of these components contains the information about different parts of a railway.

The main reason of it consists in that all channels have different angles of ultrasound impulses emission. At that only one (strictly vertical) channel receives the impulse, which is reflected from an object with the same coordinate as the receiver. For all other channels an object's coordinate differs from the receiver's coordinate and from the coordinate of the respective impulse as well, which is registered on the B-scan (Fig. 3 left).

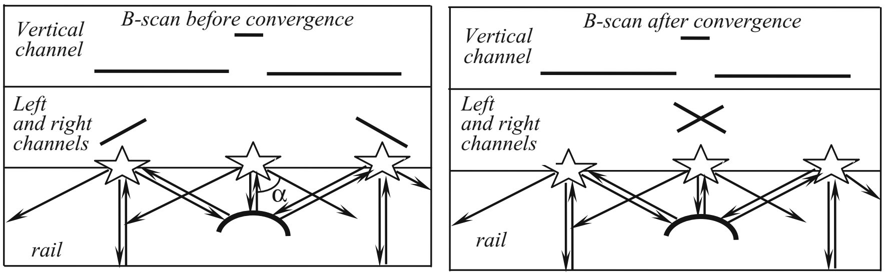


Fig. 3. Illustration of B-scan signal convergence idea

Stars in the Fig. 3 indicate some sampling flaw detector's positions during its moving from left to right. Arrows show ultrasound impulses.

The main idea of B-scan convergence consists in recomputing the impulse coordinates so as to possess the coincidence of its coordinates with the real coordinates of the objects, from which these impulses are reflected (Fig. 3 right).

Let α - be an angle to the vertical, which corresponds to the angle of impulse for some channel and let the receiver, which is situated at the distance s from the start of its moving, register a signal with some amplitude a_s in standard units of amplitude [s.u.a.] through some time τ_s in standard units of time [s.u.t.] after the pulse. Let also Δ_τ sec/s.u.t.—a constant, which characterizes the balance between seconds and flaw detector's standard units of time. The velocity of ultrasound impulse propagation under the angle α will be denoted by V^α.

Then the coordinate t of the object, which reflects the impulse, and respectively, new coordinate of the registered impulse can be determined as $t = s + (1/2)\tau_s \Delta_\tau V^\alpha \sin\alpha$.

New time delay can be computed as $\tau_t = \tau_s (V^0 / V^\alpha) \cos\alpha$, where V^0 is the velocity of vertical impulse propagation under $\alpha = 0$. The impulse's amplitude is not corrected: $a_t = a_s$.

The result of B-scans convergence is presented in Fig. 3 (right).

2.4 B-Scan Segmentation Procedure

B-scan segmentation is understood here as the selection of B-scan segments, which can contain rail flaws. The segments of interest are selected by indicating their start and end points on B-scan.

The main idea of the proposed algorithm consists in consequent passing through the defectogram and selecting the detached groups of non-zero impulses.

The proposed algorithm in pseudo-code is presented below:

```
s = 0;   // B-scan element's number
seg_start = 0;  // starting  element's number of the current segment
empty_elements = 0;  // counter of B-scan elements without registered impulses
seg_elements = 0;   // counter of current segment's elements
repeat  {
    s++;
    if  $\mathbf{X}_s$  is empty     // s-th B-scan element doesn't contain non-zero impulses
    then    empty_elements ++
    else {
            if seg_start ==0     then  {     seg_start = s;      empty_elements = 0;     }
            else  seg_elements ++;
    }
    if  (empty_elemets == M )  and (seg_start <>0)  and  (length($\mathbf{X}$)>s)
    then
        if   seg_elements  $\geq$ K
        then {        // segment is selected
            seg_end = s;
           save   seg_start    and   seg_end   for the segment;
        }
        else  {    seg_start = 0;     empty_elements = 0;       seg_elements = 0;       }
until (length($\mathbf{X}$)>s)      // till the end of B-scan signal
```

3 B-Scan Segments Comparison

3.1 B-Scan Elements Comparison

It is evident that B-scans comparison should be inevitably based on the comparison of its elements. Moreover, the quality of B-scans dissimilarity measure essentially depends on the element's dissimilarity measure.

In accordance with the Sect. 2.2, each B-scan's element for separate channel is an impulse signal and can contain $n = 0, 1, 2, \ldots$ impulses. It is impossible to compare such signals directly. So, in this paper to compare B-scan's elements we use the approach, which was initially proposed by us in [28], and consists in applying the specific mathematical model to present B-scan's elements in the form which is useful for their further comparison.

In accordance with this model each B-scan's element for a separate ultrasound channel is described by the sum of normal distributions with some standard deviation σ and mathematical expectation, which is equal to the delay of the respective impulse τ_i:

$$f(\tau|x) = \sum\nolimits_{i=1}^{n} \left(a_i \big/ \sigma\sqrt{2\pi}\right) \exp - (\tau - \tau_i)^2 \big/ 2\sigma^2. \quad (2)$$

At that, for the element without non-zero impulses $f(\tau|x) = 0$.

The graphical interpretation of the proposed model is presented at the Fig. 4 (left and center).

Let $x' = (\tau_i', a_i') \in R^2, i = 1, \ldots, n'$ and $x'' = (\tau_j'', a_j'') \in R^2, j = 1, \ldots, n''$ - two elements of the B-scan. Following [28] we will compute their dissimilarity measure as:

$$\tilde{r}(x, x'') = \sqrt{\int_{-\infty}^{\infty} [f(\tau|x') - f(\tau|x'')]^2 d\tau}. \quad (3)$$

From the geometrical point of view the dissimilarity measure (3) can be interpreted as the area within the curves of distributions, which describe the respective elements x' and x'' (Fig. 4 right).

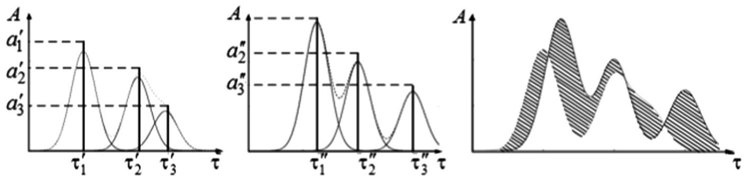


Fig. 4. The graphical interpretation of the proposed model of the B-scan's elements description for one ultrasound channel (left and center) and of their dissimilarity measure (right)

The dissimilarity measure (3) can be presented in the equivalent form:

$$\tilde{r}(x', x'') = \sqrt{\rho(x', x') + \rho(x'', x'') - 2\rho(x', x'')},$$
$$\rho(x', x'') = \sum\nolimits_{i=1}^{n'} \sum\nolimits_{j=1}^{n''} a_i' a_j'' \exp - (\tau_i' - \tau_j'')^2 / (2\sigma)^2. \quad (4)$$

At that the dissimilarity of some element $x' = (\tau_i', a_i') \in R^2,\ i = 1, \ldots, n'$ with impulses-free zero element $\phi = (0, 0)$, can be calculated by one more simple formula:

$$\tilde{r}(x', \phi) = \sqrt{\sum\nolimits_{i=1}^{n'} \sum\nolimits_{j=1}^{n''} a_i' a_j'' \exp - (\tau_i' - \tau_j'')^2 / (2\sigma)^2}. \quad (5)$$

The proposed measure guarantees that the signals, which are equal by their form, will have zero dissimilarity. The biggest dissimilarity value can be obtained when a zero signal will be compared to a signal with big amount of high amplitude impulses.

Joint using of information from different ultrasound channels is made by introducing extended dissimilarity measure, which is a linear combination of particular dissimilarity measures (4):

$$r(\mathbf{x}', \mathbf{x}'') = \sum\nolimits_{i=1}^{m} \alpha_i \tilde{r}(x'_i, x''_i), \ \alpha_i \geq 0, \ \sum\nolimits_{i=1}^{m} \alpha_i = 1, \tag{6}$$

where $\mathbf{x}' = [x'_1, \ldots, x'_m]^T$ and $\mathbf{x}'' = [x''_1, \ldots, x''_m]^T$ are two m-dimensional representations of B-scan elements in accordance with (1).

The respective way of combining information does not require any changes in the general approach to B-scan segments comparison and allows to use only one or several ultrasound channels in connection with coefficients α_i in the linear combination (6).

3.2 DTW-Based B-Scan Segments Comparison

We made the B-scan segments comparison on the basis of the adapted Dynamic Time Warping (DTW) method [25] with non standard dissimilarity measure of the signal elements (6), which allows to take into account the characteristic features of the considered problem.

Let $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_{N_\mathbf{X}})$ and $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_{N_\mathbf{Y}})$ are two multicomponent discrete signals, which represent B-scan segments for several ultrasound channels. It is required for each element of one signal to find optimal correspondent element of the other signal $T = (t_i,\ i = 1, \ldots, N_\mathbf{X})$, where $t_i \in \{1, \ldots N_\mathbf{Y}\}$ is the element's number of the signal $\mathbf{Y}$, which corresponds to the respective i-th element of signal $\mathbf{X}$, at that $t_1 = 1$ and $t_{N_\mathbf{X}} = N_\mathbf{Y}$.

We will find such optimal pairwise correspondences of elements, which posses the minimum of the optimality criterion:

$$J(\mathbf{X}, \mathbf{Y}, T) = \sum_{t=1}^{N_\mathbf{X}} \sum_{t_i} r^2(\mathbf{x}_i, \mathbf{y}_{T_I}) + \sum_{t=2}^{N_\mathbf{X}} \gamma(t_{i-1}, t_i),$$
$$\gamma(t_{i-1}, t_i) = \begin{cases} \beta|t_i - t_{i-1} - 1|, & t_i \geq t_{i-1}, \\ \infty, & t_i < t_{i-1}, \end{cases} \tag{7}$$

where $\beta > 0$ is a penalty for local stretching signal axis at signals warping.

The optimal alignment $\hat{T}$ can be found by dynamic programming procedure [29], and the obtained optimal value of the criterion $J(\mathbf{X}, \mathbf{Y}, \hat{T})$ has the meaning of B-scan segments dissimilarity:

$$d(\mathbf{X}, \mathbf{Y}) = J(\mathbf{X}, \mathbf{Y}, \hat{T}), \ \hat{T} = \arg\min J(\mathbf{X}, \mathbf{Y}, T) \tag{8}$$

Our paper [28] contains sequential algorithm of computation of this dissimilarity measure and the paper [30] contains its parallel implementation using CUDA technology.

4 Rail Flaws Recognition Experiments

Initial data for the experimental investigation has been given by Radioavionica Corporation. The initial data consists of a number of parts of B-scan signals of Russian railways, which contain 27 rail head cracks - the most frequently met rail flaws.

The filtering procedure has been applied to the initial B-scans (low amplitude probes have been removed). Then convergence and flaw localization have been made.

As a result, B-scan segments have been cut out from the initial B-scan signals $N = 324$, including 27 segments with rail flaws, which have been confirmed by experts, and 297 segments without rail flaws.

For all the obtained segment pairs the proposed dissimilarity measure $d(\mathbf{X}_i, \mathbf{X}_j),\ i, j = 1, \ldots, N$ has been computed. The Support Vector Machines (SVM) method has been used to make rail flaw recognition [31, 32], because it is one of the most convenient and effective methods of two-class pattern recognition in linear spaces. To incorporate nonlinearity into the linear decision rule of rail flaws recognition, we apply the nonlinear transformation $K(\mathbf{X}_i, \mathbf{X}_j) = \exp[-\gamma\, d^2(\mathbf{X}_i, \mathbf{X}_j)]$. It should be noticed that for the chosen value $\gamma = 0.05$ the matrix $[K(\mathbf{X}_i, \mathbf{X}_j),\ i, j = 1, \ldots, N]$ is nonnegative definite. As a result it is possible to use it as inner product matrix in SVM training procedure without any additional transformations [22].

The recognition quality has been estimated on the basis of 5folds cross-validation procedure. Mean ROC-scores (AUC values) through 5 folds for different values of the SVM's parameter C are presented in the Table 1.

For all C values the same data partitions into the folds have been used.

Table 1. Rail flaws recognition results for different values of SVM parameter C.

C	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5
AUC	0.811	0.812	0.815	0.816	0.820	0.799	0.802	0.803	0.791	0.774

In Fig. 5 ROC-curves for optimal value of SVM parameter $C = 1$ are presented.

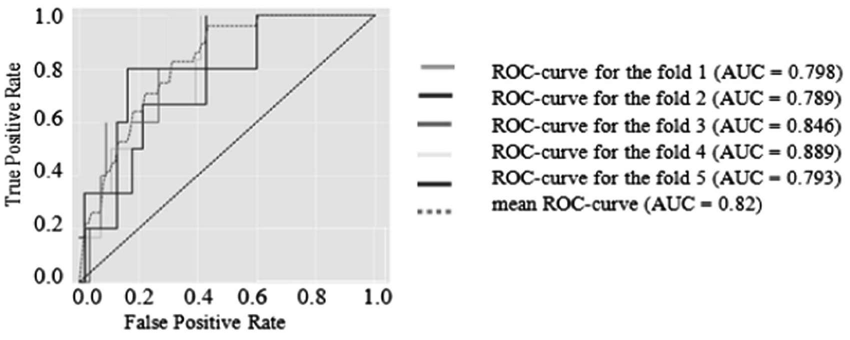


Fig. 5. ROC-curves for the optimal value of $C = 1$

5 Conclusion

Ultrasound testing is a popular technique to find some hidden rail damages. In this paper we focus on the modern Russian railway flaw detectors, such as AVICON-14, which produce the results of ultrasound testing in the form of B-scan signals. We propose a simple enough approach to the fast automatic localization of B-scan signal segments, which could contain rail flaws. For the recognition of selected segments as flaws of some kind or not flaws we apply SVM classifier jointly with DTW-based dissimilarity measure, specifically adapted by us to B-scan signals. To improve the rail flaw localization and recognition quality we preprocess B-scan signals by applying some filter and making their convergence. Fast localization procedure jointly with CUDA implementation of B-scan segments comparison possesses the possibility to process big amounts of data. The experiments have shown that all rail flaws have been localized correctly and cross-validation ROC-score = 0.82 for the rail flaw recognition has been reached.

Acknowledgements. The results of the research project are published with the financial support of Tula State University within the framework of the scientific project № 2017-66PUBL.

References

1. Jimenez-Redondo, N., Bosso, N., Zeni, L., Minardo, A., Shubert, F., Heinicke, F., Simrothhubert, A.: Automated and cost effective maintenance for railway (ACEM-Rail). Proc. Soc. Behav. Sci. **48**, 1058–1067 (2012)
2. Jemec, V., Grum, J.: Automated non-destructive testing and measurement systems for rails. In: 10th European Conference on Non-Destructive Testing, Moscow (2010). www.ndt.net/article/ecndt2010/reports/1_10_42.pdf
3. Markov, A.A., Shpagin, D.A.: Ultrasonic Rail Defectoscopy, 2nd edn, p. 284. Education – Culture, St-Petersburg (2013). (in Russian)
4. Alahakoon, S., Sun, Y.Q., Spiryagin, M., Cole, C.: Rail flaw detection technologies for safer, reliable transportation: a review. J. Dyn. Syst. Meas. Control **140**(2), 020801 (2017). https://doi.org/10.1115/1.4037295. Paper No: DS-17-1110
5. Markov, A.A., Antipov, A.G.: Magnetodynamic method for rail testing. V mire nauki **№3** (57), 66–71 (2012). (in Russian)
6. Markov, A.A., Kozyakov, A.B., Kuznetsova, E.A.: Decryption of defectogram of ultrasonic rail control. In: Practical Aid, St-Petersburg, p. 206 (2006). (in Russian)
7. Markov, A.A., Garaeva, V.S.: About acoustic contact at the area of rail bolt-on-joints. Put and putevoe hozyaistvo, № 12, pp. 15–17 (2008). (in Russian)
8. Markov, A.A., Kuznetsova, E.A., Antipov, A.G., Verevkin, A.Y.: A way of railway diagnostic. Patent for invention, № 2521095, 27 June 2014. (In Russian)
9. Lin, J., Luo, S., Li, Q., Zhang, H., Ren, S.: Real-time rail head surface defect detection: a geometrical approach. In: Proceedings of IEEE International Symposium on Industrial Electronics, pp. 769–774 (2009)
10. Gimy, J., Hyfa, N., Krishnan, R.: Rail flaw detection using image processing concepts - a review. IJERT 3(4) (2014). ISSN 2278-0181

11. Markov, A.A., Kozyakov, A.B., Kuznetsova, E.A., Shpagin, D.A.: Lost and new technologies of rail testing. Put and putevoe hozyaistvo, № 8, pp. 2–9 (2013). (in Russian)
12. Markov, A.A., Kuznetsova, E.A.: Rail defectoscopy. Signal's formation and analysis. T. 2. Defectogram interpretation, p. 332. SPb UltraPrint (2014) (in Russian)
13. Cheprasov, D.N., Malenichev, A.A., Sulimova, V.V., Krasotkina, O.V., Mottl, V.V., Markov, A.A.: Railway ultrasonic defectogram missing data recovery on the basis of semi-global alignment. In: Machine Learning and Data Mining, T. 1, № 12, pp. 1731–1751 (2015). (in Russian)
14. Shilov, M.N.: Methodological and algorithmic foundation and software for registration and analysis of defectograms at ultrasonic rail inspection, p. 153. Ph.D. thesis, St-Petersburg (2007). (in Russian)
15. Fedorenko, D.B.: Problems of automatization of multichannel ultrasonic rail signals decryption. In: Proceedings of Radioelectronnye kompleksy mnogocelevogo naznacheniya, St-Petersburg, pp. 117–120 (2011). (in Russian)
16. Heckel, T., Thomas, H., Kreutzbruck, M., Ruhe, S.: High speed non-destructive rail testing with advanced ultrasound and eddy-current testing techniques. In: Indian National Seminar and Exhibition on Non-Destructive Evaluation, NDE (2009)
17. Jiao, S.X., Wong, S.B.: Development of an automated ultrasonic testing system. In: Proceedings SPIE, vol. 5852, pp. 480–486. Nanyang Technological University, Singapore. International Society of Optical Engineers, Singapore (2004)
18. Sun, M., Lin, X., Wu, Z., Liu, Y., Shen, Y., Feng, N.: Non-destructive photoacoustic detecting method for high-speed rail surface defects. In: Proceedings of IEEE International Instrumentation and Measurement Technology Conference, I2MTC, pp. 896–900 (2014)
19. Wang, D., Lu, L., Zhang, H.J.: Speech segmentation without speech recognition. In: International Conference on Multimedia and Expo, ICME 2003, vol. 1, pp. 405–408 (2003)
20. Kehagias, A., Nidelkou, E., Petridis, V.: A dynamic programming segmentation procedure for hydrological and environmental time series. Stoch. Environ. Res. Risk Assess. **20**, 77–94 (2006)
21. Fearnhead, P.: Exact Bayesian curve fitting and signal segmentation. IEEE Trans. Signal Process. **53**(6), 2160–2166 (2005)
22. Sasan, M., Sharif, B.S.: A nonlinear variational method for signal segmentation and reconstruction using level set algorithm. Signal Process. **86**(11), 3496–3504 (2006)
23. Duin, R.P.W., De Ridder, D., Tax, D.M.J.: Experiments with a featureless approach to pattern recognition
24. Mottl, V., Dvoenko, S., Seredin, O., Kulikowski, C., Muchnik, I.: Featureless pattern recognition in an imaginary Hilbert space and its application to protein fold classification. In: Perner, P. (ed.) MLDM 2001. LNCS, vol. 2123, pp. 322–336. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44596-X_26
25. Martens, R., Claesen, L.: On-line signature verification by dynamic time-warping. In: ICPR, pp. 38–42. IEEE (1996)
26. Wang, X., et al.: Experimental comparison of representation methods and distance measures for time series data. Data Min. Knowl. Discov. 1–35 (2010)
27. Verbovoy, V.: Metrics for sound signals comparison with subject to human hearing's characteristics. In: Computer Graphic and Multimedia, № 3, pp. 2–10 (2005)
28. Malenichev, A., Sulimova, V., Krasotkina, O., Mottl, V., Markov, A.: An automatic matching procedure of ultrasonic railway defectograms. In: Perner, P. (ed.) MLDM 2014. LNCS, vol. 8556, pp. 315–327. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08979-9_24

29. Bellman, R., Kalaba, R.: Dynamic programming and modern control theory. In: Science, p. 118 (1969). Pattern Recognit. Lett. **18**(11–13), 1159–1166 (1997)
30. Zhukov, A.S., Malenichev, A.A., Krasotkina, O,V., Sulimova, V.V.: Fast algorithm of railway ultrasonic defectograms matching. In: Proceedings of XVIII International Conference on DAMDID/RCDL 2016. Analytics and Data Management in Areas with Intensive Data Using, Yershovo, 11–14 October 2016. TORUS PRESS (2016)
31. Vapnik, V.N.: Statistical Learning Theory, p. 768. Wiley-Interscience, Hoboken (1998)
32. Mottl, V., Lange, M., Sulimova, V., Yermakov, A.: Signature verification based on fusion of on-line and off-line kernels. In: 19th International Conference on Pattern Recognition, Florida, Tampa, December 2008

Social Media Sentiment Analysis Based on Domain Ontology and Semantic Mining

Daoping Wang(✉), Liangyue Xu, and Amjad Younas

University of Science and Technology Beijing, Beijing 100083, China
dpwang@ustb.edu.cn, S20171000@xs.ustb.edu.cn.com, Amjad.younas@yahoo.com

Abstract. The rapid development of social media has made more and more users express their opinions, feelings, and attitudes toward various things through different forums like Twitter, WeChat, and Weibo. However, most existing works just focus on specific product categories to construct the domain ontology, which is a quite narrow use of domain ontology. We propose a new construction of domain ontology based on the semantic features of social media. The topic of posts and opinions, also known as topic-opinion pairs, are identified with the domain ontology. The sentiment polarities are determined with the help of the given sentiment polarities. The sentiment polarity of an unknown post is calculated by the weighted average of the sentiment polarities of topics and opinions contained in the post. Preliminary results show that the application of domain ontology can effectively identify the topic-opinion pairs, and according to the known polarity of posts can effectively classify the topic-opinion pairs. The accuracy of sentiment classification is increasing.

Keywords: Domain ontology · Semantic mining · Associating mining Social media

1 Introduction

Recent years, with the booming of E-commerce and social media, numerous netizens express their views, feelings, and attitudes through Twitter, BBS, Weibo and other different ways. That emotional information of products, topics, and other valuable things not only objectively expressed their attitudes and experience but also were integrated with their own various language emotional colors and text sentiment orientation. It was increasingly being used by governments and companies to understand how the crowd think.

Semantic analysis is often called viewpoint mining or evaluation extraction, which is closely related to computer linguistics, natural language processing, and text mining. It usually refers to the data from user's subjective comment and post, using automated or semi-automatic ways to analyze and process. As a result, the opinion and sentiment orientation of individuals and groups on various topics, tasks, and other expressions could be mined. Domain ontology is extracted as a particular domain of the real world into a set of concepts and the relationship between concepts. It systematically describes the basic principles and main entities of the field, in order to realize the application and sharing of domain knowledge. Although recent years have seen a great progress in sentiment analysis and domain ontology, it still focused on specific category such as mobile phone, cars forum. By analyzing the relationship between the concepts which described in the product comments, the domain ontology for product reviews is constructed. But few papers mentioned the construction of domain ontology in social media. The immediacy of participation and dynamic communication, Because of the two main characters of social media and the relationship among the posts, it is possible to construct the domain ontology. Therefore, based on the characteristic of the posts, this paper constructs the ontology model with the semantic features in social media. In this way, the sentiment classification of a post in social media is more accurate and effective.

2 Related Work

Sentiment analysis has been extensively studied at different granularity levels. To construct affective dictionary is to use it as a prior knowledge of sentiment analysis and assist the analysis of different granularity. In addition to sentiment dictionary, ontology technology has been widely applied to the research of sentiment analysis. Many researchers try to combine domain ontology with it to improve and optimize the performance and accuracy of sentiment analysis. As a most important feature of ontology, domain ontology aims to standardize concepts and terminology in specific fields and establish a shared conceptual system between different domains, for the purpose of providing basic support for practical applications in these fields.

Marstawi [1] concluded that the sentence-level linguistic rules applied by Ontology-Based Product Sentiment Summarization could provide a more accurate sentiment analysis. Jung [2] showed that the applicability of the ontology was validated by examining the representability of 1358 sentiment phrases using the ontology EAV model and conducting sentiment analyses of social media data using ontology class concepts. Sreejith [3] used the ‘Navarasa’ ontology created by the researcher for sentiment analysis in a short story. Hu and Liu [4] used supervised sequential pattern mining method to identify and extract features or viewpoints. Wilson [5] developed an Opinion finder system, which was an automatic recognition of subjective sentences and various subjective components in a sentence (such as opinion source, emotion, direct subjective expression). Kim and Hovy [6] labeled the words that expressed subjectivity in a sentence based on a comment dictionary by manual annotation through defining a fixed-size window which was centered on subjective words. Kobayashi [7] artificially defined evaluation objects and evaluation words and described the modified relation between words and evaluation objects by using 8 common modules. Zhuang [8] defined the characteristics and viewpoints film based on WordNet, then aimed to identify features and their opinions through the dependency syntax diagram. Jakob [9] employed the conditional random field algorithm to extract the feature and opinions. Tan [10] tried to apply behavioral relation data on social media to user-level sentiment analysis according to the idea that two users with mutual relationships were likely to hold the same view. Go [11] attempted three machine learning methods, namely Naïve Bayes, support vector machines and maximum Entropy in text sentiment orientation in Twitter, they conclude that the applicability of machine learning model in the sentiment analysis.

On the recognition of the point of view, Alexander [12] considered naive Bayes classifier to identify the point of view based on characteristics extracted by POS tagging and N-gram. Barbosa [13] used the subjectivity of the words, the polarity of words and negative words as characteristics to classifier the subjective and objective nature of Weibo posts. Davidiv [14] extracted tags and emoticons from Twitter as training sets and used an advanced KNN algorithm to classify sentiment on Weibo posts. Jain [15] combined LDA with data expansion method for data preprocessing and made sentiment evaluation by Naive Bayes dual training and Domain prediction algorithms.

Different from most existing studies which concentrate on concrete product, this paper extracts feature entries based on the semantic features of social media and constructs domain ontology. In this study, the emotion value of topic-opinion pairs is weighted by semantic context so that sentiment classification could be more accurate and effective.

3 Construction of Domain Ontology in Social Media

The traditional domain ontology is generally based on collecting characteristics of terms. However, due to many characteristics in a post on social media, such as short texts, incorrect writings, disorderly sentence structure, the keywords that are really relevant to the central idea may be less than expected, and the word frequency may not be the highest. It means that the accuracy of defining topics and opinions of a post is low, and mining semantic information in the posts is an effective way to solve this problem. According to the characteristics of language habits and oral expressions, the content of a post can be generalized not only two main parts topics and comment but also five elements: time, location, object, event, and opinion. It means that to some extent the first four elements can describe the topic. In this paper, the author used these four elements as the foundation of the domain ontology in social media.

3.1 Extracting Topic Characteristics of Terms

Extracting topic terms is the foundation of the constructions of the domain ontology. Those existing methods mostly focus on English comment and product comment. Because of the grammatical nonstandard, semantic ambiguity and subject missing, the characteristics of Chinese comment increase the difficulty of sentiment classification. Therefore, the representation model of domain ontology is proposed which regards the definition of topic characteristics of terms.

Definition 1. *A topic term can be defined as <Time, Location, Object, Event>.*

Due to the colloquial characteristics of social media, there will be a lot of irregular and vague expressions such as “tomorrow”, “yesterday” in a post, and the formulations are various, for example, “3:30 p.m.”, “half past three in the afternoon”, “three thirty”, those irregular expressions should be standardized. The standard form of time is yyyy-MM-dd-HH-mm-ss. Similarly, the expression of location needs to be standardized as Room, Unit, Building, Road, District, City, Province, and Country.

After standardization, the next step is to mine frequent itemsets with association rules in content. First, all nouns and verbs are extracted from the post to form two itemsets. After mining these two itemsets with association rule, the frequent 1-itemsets and frequent 2-itemsets are found. By filtering out frequent 2-itemsets which cannot form phrases and that itemsets unrelated to time, location, object, and event, which identify the category of a post can be formed.

After acquiring a set of feature words, by means of the point mutual information (*PMI*) between the feature words, the semantic common frequency of the word is expressed. The higher frequency of the two words in the post shows the higher correlation. Hence, the *PMI* defined as follows:

$$PMI = \log \frac{p(word_1 \,\&\, word_2)}{p(word_1) * p(word_2)} \tag{1}$$

We take $PMI(word_1, word_2)$ for *a* frequency of two words' co-occurrence, $P(word_1)$ and $P(word_2)$ shows frequency of one word appearing respectively. $word_1$, $word_2$ can be taken from the same category of feature words and different category of feature words. After calculating *PMI*, combined those feature words of high *PMI* value to the topic term.

Definition 2. *Domain Ontology uses a 2-tuple representation,* $O = \, <C, R>$ *, C represents for the topic item, R represents for the relations between two topic items.*

The topic item is represented by $C = \, <ID, Topic, List>$. *ID* represents the only number of the topic item, *Topic* represents the descriptive words of this topic item, *List* represents for the synonyms of this topic item. The relation between two topic items is represented for $R = \, <T, R(C_1, C_2)>$. We take *T* for describing three semantic relations: part of, relevant with, and irrelevant with, C_1 and C_2 represent for two topic items.

3.2 Clustering Algorithm of Topic Items

After preprocessing of the post (including syntactic structure, deactivating words and POS tagging), we need to preserve nouns and verbs which are related to time, place, object, and event. In the training sets, we extract four categories which are composed topic items, the *Time* characteristic in *Topic i* items are represented for vector $T_i = \{t_1, t_2 \ldots t_n\}$, the *Location* for vector $L_i = \{l_1, l_2, l_3 \ldots l_n\}$, the *Object* characteristic for vector $O_i = \{o_1, o_2, o_3 \ldots o_n\}$, the *Event* characteristic for vector $E_i = \{e_1, e_2, e_3 \ldots e_n\}$.

The weights of word t_i in *Time* characteristic in the topic item is calculated as follows:

$$\eta_T(t_i, A_i) = \frac{tft(t_i)}{\max tf(A_i)} \log_2 \frac{M}{df(t_i)} \tag{2}$$

We take $tf(t_i)$ for the frequency of t_i appearing in a post A_i; $df(t_i)$ for the number of post which contained t_i; Function max $tf(A_i)$ for the maximum word frequency in the post A_i; *M* for the number of post in the training set. To make the weight between [0, 1], $\eta_t T(t_i, A_i)$ is defined as follows:

$$\eta_t(t_i, A_i) = \frac{\eta_T(t_i,) - \min\eta_T}{\max\eta_T - \min\eta_T} \tag{3}$$

After calculating the average amount of $\eta_T(t_i, A_i)$, in the post A_i is calculated as follows:

$$\eta_T(A_i) = \frac{\sum_{i=1}^{M} \eta_t(T_\mathrm{i}, A)}{M} \tag{4}$$

Similarly, we get the formulation for the weights of *Location* characteristic $\eta_l(A)$, the weights of *Object* characteristic $\eta_o(A)$ and the weights of *Event* characteristic $\eta_e(A)$.

The similarity between two topic terms could be calculated by vector space model, vector A_i, and A_j, $A_i = \{T_i, L_i, O_i, E_i\}, A_j = \{T_j, L_j, O_j, E_j\}$, the formulation between A_i and A_j is shown as follows:

$$Sim(A_i, A_j) = \eta_T * Sim(T_j, T_j) + \eta_l * Sim(L_\mathrm{i}, L_j) + \eta_O * Sim(O_i, O_j) + \eta_e * Sim(E_i, E_j) \tag{5}$$

The similarity between the *Time* vector in post A_i and that in post A_j is calculated as follows:

$$Sim(T_i, T_j) = \frac{\sum_{k=1}^{n} \eta_k(T_i) * \eta_k(T_j)}{\sqrt{(\sum_{k=1}^{n} \eta_k^2(T_i)) * (\sum_{k=1}^{n} \eta_k^2(T_j)}} \tag{6}$$

We take $\eta_k(T_i)$ for the kth word in the *Time* vector of post A_i, $\eta_k(T_j)$ for the kth word in the *Time* vector of post A_j.

Similarly, we get the formulation of similarity between the *Location* vector and the *Object* vector.

$$Sim(L_i, L_j) = \frac{\sum_{k=1}^{n} \eta_k(L_i) * \eta_k(L_j)}{\sqrt{(\sum_{k=1}^{n} \eta_k^2(L_i)) * (\sum_{k=1}^{n} \eta_k^2(L_j))}} \tag{7}$$

$$Sim(O_i, O_j) = \frac{\sum_{k=1}^{n} \eta_k(O_i) * \eta_k(O_j)}{\sqrt{(\sum_{k=1}^{n} \eta_k^2(O_i)) * (\sum_{k=1}^{n} \eta_k^2(O_j))}} \tag{8}$$

Based on the Hownet a semantic knowledge resource, calculation of the *Event* vector similarity is the maximum of similarity between their original semantic meanings.

After acquiring the *Topic* items composed of the *Time* vector, the *Location* vector, the *Object* vector, and the *Event* vector, the advanced K-Nearest neighbor algorithm is used to do the cluster analysis (See Fig. 1).

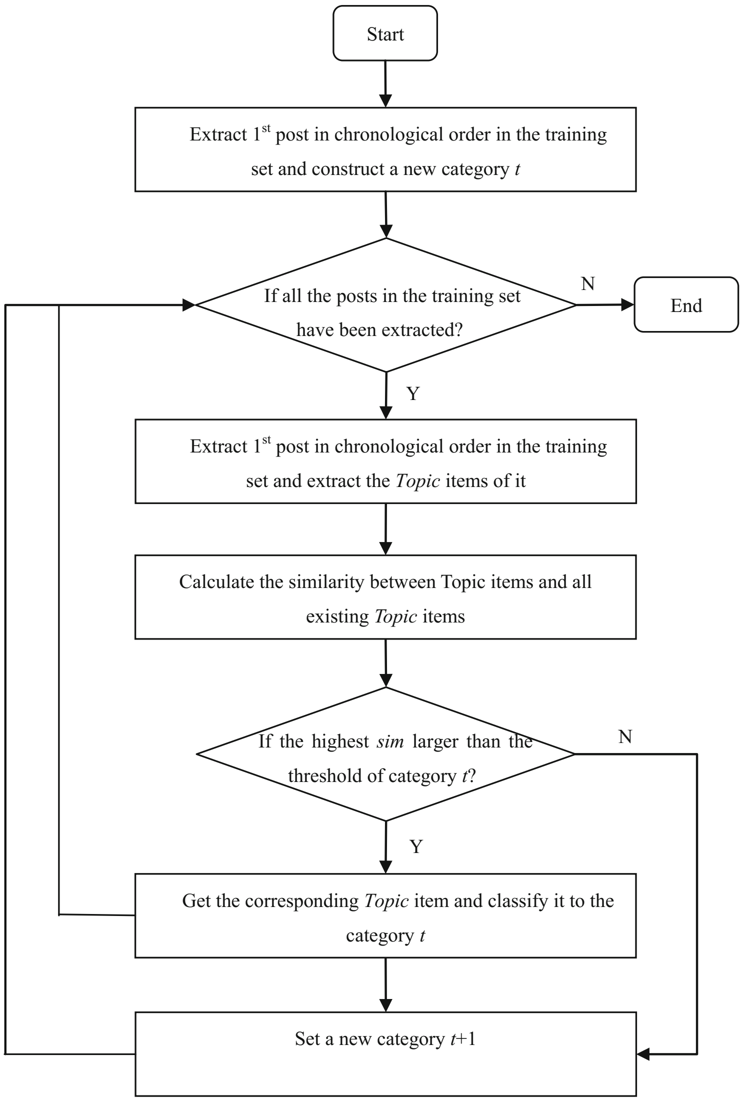

Start
Extract 1st post in chronological order in the training set and construct a new category t
If all the posts in the training set have been extracted?
N
End
Y
Extract 1st post in chronological order in the training set and extract the Topic items of it
Calculate the similarity between Topic items and all existing Topic items
If the highest sim larger than the threshold of category t?
N
Y
Get the corresponding Topic item and classify it to the category t
Set a new category t+1

Fig. 1. Clustering algorithm of *Topic* items

4 Identification of Topic-Opinion Pairs Based on Domain Ontology

After the construction of the domain ontology in social media, the next step is to extract the Opinion from the post based on semantic mining.

4.1 Identification of Subjective Sentence

The identification of the *Opinion* is the process of classification of sentences building on sentence structure and POS. Using paper [16] and analysis of numerous posts for reference, the three main categories of feature words in the subjective sentence are summarized: sentiment words, modal particle and asserted words.

Sentiment Words. The subjective sentence in a post always contains the standpoint and viewpoints of the author and a strong individual initiative, so the sentiment words could be one of the characteristics of the subjective sentence.

Asserted Words. For example, "claim", "blame", "announce", these words' appearance could be seen as the strong possibility of the subjective sentence.

Modal Particle. The emotional punctuations such as "!", "?" and emoticons could express the individual initiative of the author, as well as the Chinese only model particles "吗", "呢", "吧".

4.2 Extraction of Key Subjective Sentence and Relations

The posts in social media are informative but semantic fuzziness, according to the automatic summarization, we extract the keywords for subjective sentence *B*. Based on the topic relevance and three important property, decide the key subjective sentence. After weighted summation of the topic relevance, position property, sentiment property, and keyword property, we take the corresponding sentence with the highest value for key subjective sentence. The formulation is shown as follows:

$$\begin{aligned} key\_Sentence = {} & \lambda_1 * sim(s_i, c_i) + \lambda_2 * keywords(s_i) + \lambda_3 * position(s_i) + \lambda_4 \\ & * senti\_words(s_i) \end{aligned} \tag{9}$$

We take λ_1, λ_2, λ_3, and λ_4 for the corresponding weight for the four elements.

$Sim(S_i, C_i)$ represents the relevance between the subjective sentence and topic. Combining the topic terms *C* with the subjective sentence, we get a sequence of topic-opinion which are separated by "," and calculate the similarity in pieces. According to the term frequency-inverse document frequency, we obtain $sim\left(w_{i,k}, c_{i,k}\right)$ as follows:

$$sim(w_{i,k}, c_{i,k}) = \sum_{c \in C} tf_p(w_{i,k}) * tf(p_{i,k}) * \log_2 \frac{N}{pf(w_{i,k})} \tag{10}$$

Where $w_{i,k}$ denotes the feature words k in the subjective sentence i, $c_{i,k}$ denotes the feature word k contained in the topic terms c_i, $p_{i,k}$ denotes the combination of $w_{i,k}$ and $c_{i,k}$, $tfp\left(w_{i,k}\right)$ denotes the frequency of $w_{i,k}$ in $p_{i,k}$, $f(p)$ denotes the frequency of p in the whole post, $pf\left(w_{i,k}\right)$ denotes the total number of phrases which included $w_{i,k}$, N denotes the total number of phrases in the post.

Marked the highest value of $sim\left(w_{i,k}, c_{i,k}\right)$ as $sim(S_i, C_i)$ in the formulation (11) and the corresponding feature words as the indicator d_{si} of opinion$_{\mathrm{i}}$.

$$sim(S_i, C_i) = \max(sim(w_{i,k}, c_{i,k})) \tag{11}$$

$Position(s_i)$ presents for the different part of the speech. People are willing to speak his mind at the beginning of a speech and summarize at the end, so it attaches great importance the opening phrase and the end of the statement. The formulation (12) presents $position(s_i)$.

$$position(s_i) = [i - \frac{num(s_i)}{2}]^2 + 1 \tag{12}$$

Where $num(s_i)$ denotes the total number of sentences in a post, the constant 1 is to confirm every sentence in different position has a positive score. From the formulation, it can be seen that the function is a pointing-up parabola which axis of symmetry is at the central position. It confirms that the first and ending sentence has the more location advantages than others.

$keywords(s_i)$ identifies those words which are general and set the tones, such as “anyway”, “in a word”. If a sentence includes these words, the possibility of being a key-opinion sentence is increasing. The formulation of $keywords(s_i)$ is shown as follows.

$$keywords(s_i) = \sum_{k=1}^{num(w_{i,k})} keyword(w_{i,k}) \tag{13}$$

Where $keywords\ \left(w_{i,k}\right)$ denotes that whether word $w_{i,k}$ is a summary word or not. If it is, $keywords\ \left(w_{i,k}\right)$ is 1, otherwise $keywords\ \left(w_{i,k}\right)$ is 0.

$senti\_words(s_i)$ identifies the sentiment orientation of sentences which is shown as follows.

$$senti\_words(s_i) = \frac{\sum_{k=1}^{num(w_{i,k})} polarity(w_{i,k})}{num(w_{i,k})} \tag{14}$$

Where $polarity\ \left(w_{i,k}\right)$ denotes that whether word $w_{i,k}$ has sentiment orientation or not. If it is, $polarity\ \left(w_{i,k}\right)$ is 1, otherwise $polarity\ \left(w_{i,k}\right)$ is 0.

5 Post Sentiment Analysis Based on Domain Ontology

In order to make the sentiment analysis more targeted, we set up rules for matching the post with domain ontology.

Rule 1. The topic items and the opinion indicator are both contained in the post, match the post with the domain ontology.

Rule 2. Only topic items are included, not the subjective sentence and opinion indicator, just assume that this post is an objective statement and without any individual initiative. Therefore, filter this post and the extracted topic item does not match any subjective sentences.

Rule 3. Only subjective sentence is included, not the topic items. There is always be ignoring of subject, incidents and time in speech, for example, "We all should engrave what happened on May 5th, 2012 on our mind", we need to identify those implicit topics. Assume that only subjective sentence Bi and topic item i without the object i, use the extracted topic $i < C_i, S, d_{si} >$ as the prior knowledge to calculate the similarity. The missing object i is the corresponding O_i, S of the highest $sim(C_i, S - O_i, S, (C_i - O_i,), d_{si}, B_i)$. Similarly, we could get the implicit *Time*, *Location*, and *Event* in the post.

According to the domain ontology and sentiment dictionary, we get the topic item, subjective sentence and sentiment words in the post. The next step we need calculate the sentiment orientation of the subjective sentence. The level of affect intensity is $Q = \{q_1, q_2, q_3 \ldots q_i \ldots q_n\}$. When calculate the sentiment value, not only the sentiment polarity and affect intensity should be considered, but also the adverb of degree and negation words appeared closely should also be considered. Therefore, the sentiment value of a sentence is represented as follows.

$$senti\_couple(j) = \sum_{k=1}^{num(C_k)} sw_k * (-1)^p * d(adv) \tag{15}$$

Where $senti\_couple(j)$ denotes the sentiment value of the topic-opinion pair j, sw_k denotes the sentiment value of sentiment words k, that is the product of sentiment polarity and affect intensity, p denotes the number of negation words appeared closer to the sentiment word k, $d(adv)$ denotes adjustment for adverb of degree appeared closer to the k.

The sentiment value of a post t is represented as follows.

$$senti\_post(t) = \frac{\sum_{i=1}^{num(senti\_(couple))} senti\_couple(i)}{num(senti\_couple(i))} \tag{16}$$

Where $num(senti\_(couple))$ denotes the number of topic item included in post t.

Due to the strong interactivity of social media, generally every post has comments, likes, and reposts. To some extent, these behaviors show the affect intensity of posts. According to Paper [17], all users can be divided into five groups by their participation and engagement (see Fig. 2): inactive user R_1, sidelines R_2, participator R_3, criticizer R_4, and key opinion leader R_5.

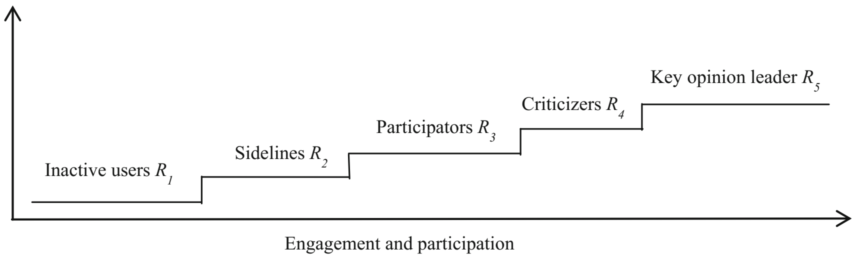


Fig. 2. Five groups of users divided by engagement and participation

Based on the different categories and these interactive behaviors, the sentiment value of a post (t, R_i) can be weighted from *senti_(post)* as follows.

$$post(t, R_i) = (1 + 0.1 * i) * senti\_post(t) * (1 + 0.05 * num(comment) + 0.02 * num(likes) + 0.05 * num(repost)) \quad (17)$$

Where R_i denotes the group which the author belonged to, i = 1, 2, 3, 4, 5, 6, *num* (*comment*) denotes the number of comments, *num*(*likes*) denotes the number of likes, *num*(*repost*) denotes the number of reposts.

6 Experiments and Results

The experimental steps are designed as follows: first, collect experimental corpus in a famous auto forum (AutoHome), and remove inactive words and POS tagging within 2747 posts. The experimental posts are divided into training set and test set. The former is used to construct the domain ontology and compute the polarity of the topic-opinion pairs, while the latter is used to evaluate the validity of the method. Then, the semantic computation of the training set is carried out and the domain ontology of social media is constructed. Based on it, the topic-opinion pairs of the test set are identified. Then, according to the positive and negative posts in the training set, the sentiment polarity value of the topic-opinion pairs is calculated and the sentiment classification is obtained. Finally, the experimental results are compared with the manual tagging results in the test set, and the method of this car model is evaluated.

Due to limited time, the experiment is still in progress. Preliminary results show that the application of domain ontology can effectively identify the sentiment polarity of posts, which are shown in Table 1.

Table 1. Preliminary results compared with manual tagging

Methods (total: 2747 posts)	Positive	Neutral	Negative
Manual tagging	984	1304	459
This paper's method	1308	1094	345
Only use sentiment dictionary	2402	48	297

7 Discussion

A post in social media may contain both positive and negative polarity, and sentiment words may change with the context. The existing research uses the context-independent sentiment classification method, or only uses the simple and empirical method to analyze context and evaluate opinions. These strategies all have some deficiencies, resulting in the lower accuracy of sentiment identification.

Therefore, this research proposes a new sentiment classification method based on social media domain ontology. Compared with the existing methods which mostly focus on the characteristics of the products through the sentiment ontology, this paper employs the semantic features and semantic relation in posts to identify the topic-opinion pairs with the social media domain ontology. Finally, the sentiment classification of each post is obtained according to the user classification and interactive characteristics of social media. In this paper, there are some deficiencies, such as sentiment classification, the neutral posts which are often ignored should be mining out the implied sentiment orientation. There is a main limitations in the results. The number of records of data was limited, only including the data from auto industry. The error of sentiment polarity value may be caused by the lack of data and data sparseness.

References

1. Marstawi, A., Sharef, N.M., Aris, T.N.M.: Ontology-based aspect extraction for an improved sentiment analysis in summarization of product reviews. In: International Conference on Computer Modeling and Simulation, pp. 100–104. ACM, Canberra (2017)
2. Jung, H., Park, H.A., Song, T.M.: Ontology-based approach to social data sentiment analysis: detection of adolescent depression signals. J. Med. Internet Res. **19**(7), 259 (2017)
3. Sreejith, D., Devika, M.P., Tadikamalla, N.S., Mathew, S.V.: Sentiment analysis of English literature using rasa-oriented semantic ontology. Ind. J. Sci. Technol. **10**(24), 1–11 (2017)
4. Hu, M., Liu, B.: Mining opinion features in customer reviews. In: National Conference on Artificial Intelligence, pp. 755–760. AAAI Press, San Jose (2004)
5. Wilson, T., Hoffmann, P., Somasundaran, S.: OpinionFinder: a system for subjectivity analysis. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, HLT/EMNLP, Canada, pp. 347–354 (2005)

6. Kim, S.M., Hovy, E.: Extracting opinions, opinion holders, and topics expressed in online news media text. In: Proceedings of the Workshop on Sentiment and Subjectivity in Text, pp. 1–8. ACL, Sydney (2006)
7. Kobayashi, N., Inui, K., Matsumoto, Y., Tateishi, K., Fukushima, T.: Collecting evaluative expressions for opinion extraction. In: Su, K.-Y., Tsujii, J., Lee, J.-H., Kwong, O.Y. (eds.) IJCNLP 2004. LNCS, vol. 3248, pp. 596–605. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30211-7_63
8. Zhuang, L., Jing, F., Zhu, X.Y.: Movie review mining and summarization. In: Proceedings of the 15th ACM International Conference on Information and Knowledge Management, pp. 43–50. ACM, New York (2006)
9. Jakob, N., Gurevych, I.: Extracting opinion targets in a single and cross-domain setting with conditional random fields. In: Conference on Empirical Methods in Natural Language Processing, Cambridge (2010)
10. Tan, C., Lee, L., Tang, J.: User-level sentiment analysis incorporating social networks, pp. 1397–1405. ACM, San Diego (2011)
11. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. Cs224n Project Report, pp. 1–12 (2009)
12. Alexander, P., Patrick, P.: Twitter as a corpus for sentiment analysis and opinion mining. In: Proceedings of International Conference on Language Resource and Evaluation, Lisbon, pp. 1320–1326 (2010)
13. Barbosa, L., Junlan, F.: Robust sentiment detection on Twitter from biased and noisy data. In: Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, pp. 36–44 (2010)
14. Davidiv, D., Tsur, O., Rappoport, A.: Enhanced sentiment learning using Twitter hashtags and smileys. In: Proceedings of the 23rd International Conference on Computational Linguistics, Beijing, pp. 241–249 (2010)
15. Jain, J.P., Awati, J., Alzende, H.: Opinion analysis of text on the basis of three domain classification. In: International Conference on Automatic Control and Dynamic Optimization Techniques. IEEE, Pune (2017)
16. Ding, S.C., Ma, M.R., Li, X.: Study of subjective sentence identification oriented to Chinese micro blog. J. China Soc. Sci. Tech. Inf. **33**(2), 175–182 (2014). (in Chinese)
17. Ramesh, S., Duerson, D., Ephraims, T.: A Managerial Perspective on Analytics. Chinese Machine Press, Beijing (2015)

Mining Associations in Large Graphs for Dynamically Incremented Marked Nodes

Anshul Rai(✉), Zackary Crosley(✉), and Srivignessh Pacham Sri Srinivasan(✉)

School of Computing, Informatics and Decision Systems Engineering, Arizona State University, Tempe, AZ 85281, USA
{arai10,zcrosley,spachams}@asu.edu

Abstract. The edges between nodes of a graph describe some sort of relationship between the two nodes. In this paper, we would like to efficiently determine the relationship between specific nodes of importance, which we call marked nodes, in a large graph. These relationships obtained must be optimal, which requires us to segregate the marked nodes from the less important nodes and group them together using partitioning algorithms. We introduce an improved algorithm which allows for the efficient addition of new marked nodes to a partition without rerunning the algorithm on previously marked nodes.

Keywords: Marked nodes · Partitioning · Graph theory

1 Introduction

The edges between nodes of a graph describe some sort of relationship between the two nodes. In this paper, we would like to efficiently determine the relationship between specific nodes of importance, which we call marked nodes, in a large graph. These relationships obtained must be optimal, which requires us to segregate the marked nodes from the less important nodes and group them together using partitioning algorithms. We introduce an improved algorithm which allows for the efficient addition of new marked nodes to a partition without rerunning the algorithm on previously marked nodes. In modern computer applications, graphs are needed to represent numerous types of data. Graphs represent our social networks in social media applications, navigation within a website, and any other data which focuses on the relationship between items. Understanding the association between elements is crucial to the development of algorithms which use these relationships to make decisions without human involvement. These algorithms recommend new connections on social media, suggest new products for purchase by a customer, serve relevant ads to users, and many other personalization strategies characteristic of modern software products. In these learning and intelligent algorithms, partitioning [1–3] is often a crucial component in understanding the similarities between unique elements.

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 40–48, 2018.
https://doi.org/10.1007/978-3-319-96136-1_4

Defined partitions can be used to quickly designate what elements belong to a category or have certain properties at the expense of significant preliminary computation. Unfortunately, the processes used to create partitions are often extremely expensive, and in many dynamic programs, partitions must be regularly reevaluated. In an online marketplace with hundreds of purchases per day, partitions which are created one day may quickly fall out of relevance. New products may be added whose relationship to other nodes needs to be evaluated, or user purchase trends may change, thus altering our partitions. In most partition algorithms updating means reevaluating from the beginning, requiring significant overhead and repeated computation.

Our research focuses on how partitions can be updated with limited computation when a new node is added. Specifically, given a partition that was created from a set of marked nodes our research tries to answer how one can add a newly marked node to that partition. As an example, suppose the relationship between products that have been often bought in tandem has been established. If several new products begin to be purchased alongside those already partitioned products, how can they be added to that partition for providing product recommendations without having to recompute the subgraph?

Several algorithms have been created to form partitions in a graph [4–8] which contain a certain set of marked nodes. These algorithms create a subgraph known as a minimum spanning tree or an arborescence [9,10]. Dot2Dot algorithm [11] efficiently finds the minimum arborescence of a graph with marked nodes. However, to add another marked node to this subgraph requires the algorithm be rerun with the union of the sets of marked nodes. This leads to an unnecessary reiteration of operations with a large time complexity. No algorithm could be found which allows for the addition of new nodes to a partition without the reproduction of all computation steps.

Generally, algorithms similar to Dot2Dot have five approaches. They all are used to find the set of trees with minimum total cost on marked nodes.

- Finding Bounded Path Length: Finding shortest path between terminal nodes using BFS-like expansion till the threshold path length, $log|V|$ (where $|V|$ indicates the total number of vertices in a graph), is reached.
- Connected Components: The algorithm detects connected components and puts all directly connected terminal nodes in one group. Nodes which do not satisfy both properties are placed into separate groups.
- Minimum Arborescence: The algorithm uses the transitive closure graph of terminals to find the minimum cost graphs. The algorithm calculates bounded path lengths and connected components.
- 1-Level Tree: The algorithm returns a forest of Steiner trees by building a level 1 tree from the transitive closure graph by considering each terminal node as root to find the shortest path on the transformed graph.
- 2-Level Tree: This algorithm generalizes 1-level tree algorithm to k-level trees. This is achieved by reducing the cost of 1-level trees successively in each step.

In this paper, we introduce an algorithm which allows for the quick addition of new marked nodes to a partition. We illustrate the success of our algorithm by providing example datasets and the corresponding output. Finally, we present the relative run times between our algorithm and the original Dot2Dot algorithm to convey the reduction in time complexity.

2 Developing the Algorithm

2.1 Problem Complexity

Suppose we have a large graph in which some of the nodes are marked (Fig. 1). We want to find the relationship between these marked nodes and to determine whether they are close to each other in the graph or if they are forming separate groups. Moreover, the relationships chosen must be optimal, that is, it should neither have too many connections nor too few connections.

● Normal Node ● Marked Node ● Newly Marked Node

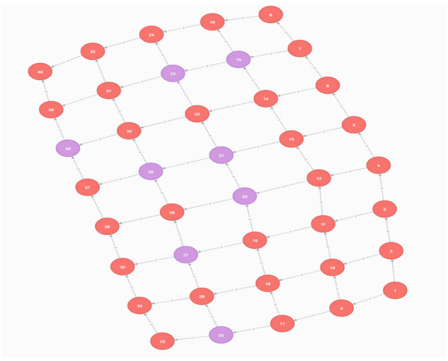


Fig. 1. Simple graph with marked nodes Set

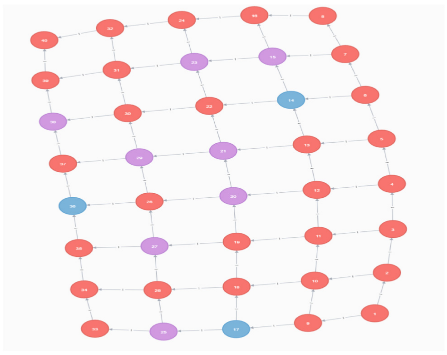


Fig. 2. Simple graph with new marked nodes Added

For this, we need to segregate these marked nodes from the rest and group them using partitioning algorithms similar to Dot2Dot which constructs good connection paths while separating the distant nodes. If a new set of marked nodes is added (Fig. 2), how can the relationship of these newly marked nodes to the existing marked nodes be efficiently determined?

It has been shown that the average performance of Dot2Dot - Minimum Arborescence algorithm is better for simulations on real-life problems [11]. Thus, Dot2Dot - Minimum Arborescence was used as the base algorithm for implementation of our improvements.

Suppose we have n marked nodes and we compute minimum arborescence (Fig. 3) using the Dot2Dot algorithm, the algorithm has a running time complexity of $O(n^2)$ [11]. Now, suppose m additional marked nodes are added. If the Dot2Dot - Minimum Arborescence algorithm is used, generating a new partition as seen in Fig. 4 below, then the running time complexity would be $O((n+m)^2)$. To improve this time complexity, we propose a solution which only explores the bounded paths for the newly marked nodes m nodes and reuses the existing bounded paths between the n marked nodes. This method reduces the time complexity from $O((n+m)^2)$ to $O(m \times (n+m))$ with respect to the number of nodes, and an exponential to polynomial reduction in the number of edges to be considered.

● Normal Node ● Marked Node ● Newly Marked Node

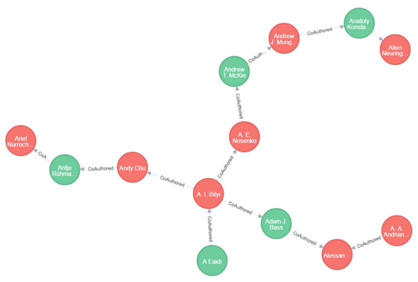


Fig. 3. Sample partition of marked nodes

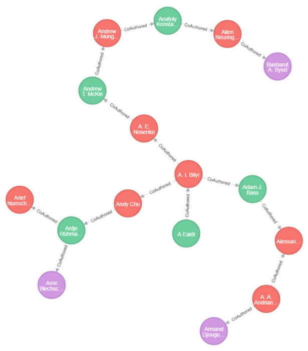


Fig. 4. Partition of marked nodes with newly added nodes

2.2 Proposed Method for Addition of New Marked Nodes

Consider an undirected graph G having vertices V and edges E such that $G = (V, E)$. Suppose we have a set M of marked nodes, also called terminals, such that $M \subseteq V$. Running the Dot2Dot algorithm outputs the Minimum Spanning Tree(T) which contains the paths that optimally describe the relationships between terminals. Since we assume that the new marked nodes are selected dynamically, the proposed algorithm must be an **online algorithm**. Hence, our algorithm incorporates each new marked node iteratively to arrive at an optimal solution.

To find the optimal relationship between one newly marked node and marked nodes in T we utilize a key property of Graph Theory: Given a Minimum Spanning Tree (T^*) and a new node (n^*), let E^* be the set of edges in T^* originating from n^*. The Minimum Spanning Tree that contains nodes of T^* and n^* is guaranteed to only contain edges from E^*. This reduces the search space of possible paths that must be considered from **exponential** (with respect to edges of marked nodes) to **polynomial**.

The steps we take once we have a Minimum Spanning Tree(T) and a set of new marked nodes(N) are as follows:

1. Take the first marked node from N, let us call it N_i.
2. Compute Paths of N_i to all nodes in T. We use a Breadth First Search from N_i to compute paths of length $\leq log|V|$. The paths are computed the same way as shown in step 2 of Dot2Dot - Minimum Arborescence algorithm. We store all paths from N_i to nodes which are in T, both marked and unmarked.
3. Find Minimum Spanning Tree of paths in T and the paths computed in the previous step. This is done since there may be multiple paths from N_i to nodes in T, so we compute the Minimum Spanning Tree once again in order to only keep the optimal relationships between the nodes. This tree obtained is our new Minimum Spanning tree, T.
4. Remove N_i from the set of new marked nodes, N.
5. Repeat steps 1 to 4 till N is non-empty.

3 Results

In this section, the proposed method is tested on sample data. We aim to test the time complexity and cost performance of the algorithm. To do so, we choose the Amazon product co-purchasing network [12] to study the performance of our approach. The network was collected by crawling the Amazon website. It is based on the 'Customers Who Bought This Item Also Bought' feature of the Amazon website. If a product i is frequently co-purchased with product j, the graph contains a directed edge from i to j.

The dataset statistics are shown in the Table 1. The Fig. 5 shows the visualization of first 50 nodes of the network.

Table 1. Dataset statistics.

Dataset statistics	
Nodes	262111
Edges	1234877
Nodes in largest WCC	262111 (1.000)
Edges in largest WCC	1234877 (1.000)
Nodes in largest SCC	241761 (0.922)
Edges in largest SCC	1131217 (0.916)
Average clstering co-efficient	0.4198
Number of triangles	717719
Fraction of closed triangles	0.09339
Diameter (longest shortest path)	32
90-percentile effective diameter	11

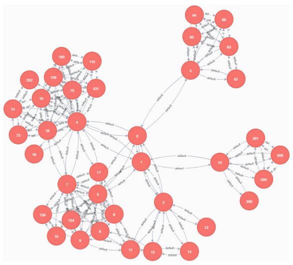


Fig. 5. Visualization of fifty nodes in data Set

We take an input graph that contains the first 1021 nodes of the network. Suppose there are 10 marked nodes initially and we incrementally mark 400 additional nodes. The standard algorithm is used to compute the initial 10 marked nodes and our proposed algorithm is used to incrementally add nodes.

Figure 6 shows the time performance in seconds for the proposed algorithm for the addition of each node. The proposed algorithm grows almost linearly with respect to the number of marked nodes. The proposed algorithm finishes

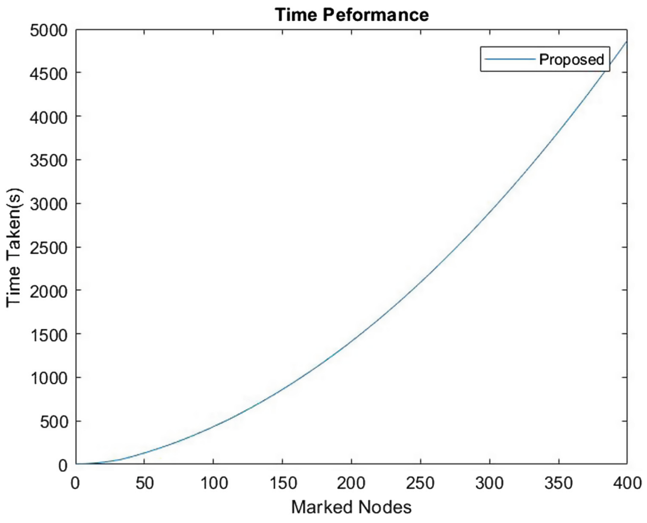


Fig. 6. Time performance of proposed algorithm by number of nodes

its computation in approximately 81 min. Figure 7 below shows a comparison between the execution time of Dot2Dot and our proposed modification.

Note the substantial difference between the standard algorithm and the proposed near the axis in Fig. 7. The standard algorithm has an exponential growth with respect to the number of edges between marked nodes and takes approximately 5544 min (3.85 days). Our proposed approach finishes its computation with a 98.54% reduction in runtime.

Figure 8 shows the comparative cost of the proposed algorithm with the standard Dot2Dot - Minimum Arborescence algorithm, where cost is the weighted sum of all the edges in the graph. From the cost performance, we can interpret that the proposed algorithm produces extremely competitive results despite taking less computation time.

Since graph partitioning is NP-complete achieving equivalent output between the two graph algorithms is not guaranteed, however, the near-equal cost suggests that the partitions created by both algorithms are equally valid. Thus, we can conclude that the proposed algorithm provides an equivalently optimal solution like the Dot2Dot algorithm, but does it in a significantly more efficient way. A sample partitions from our tests, each beginning with ten originally marked nodes and adding thirty newly marked nodes, can be seen below in Figs. 9 and 10, which use the standard Dot2Dot - Minimum Arborescence and proposed algorithm respectively.

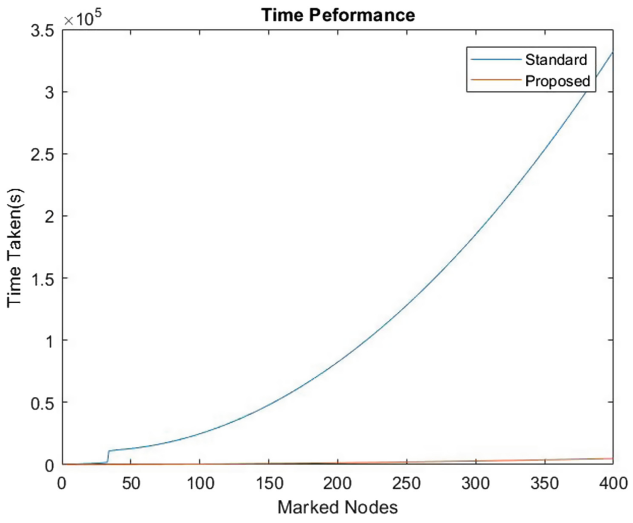


Fig. 7. Dot2Dot (Standard) performance compared to proposed algorithm

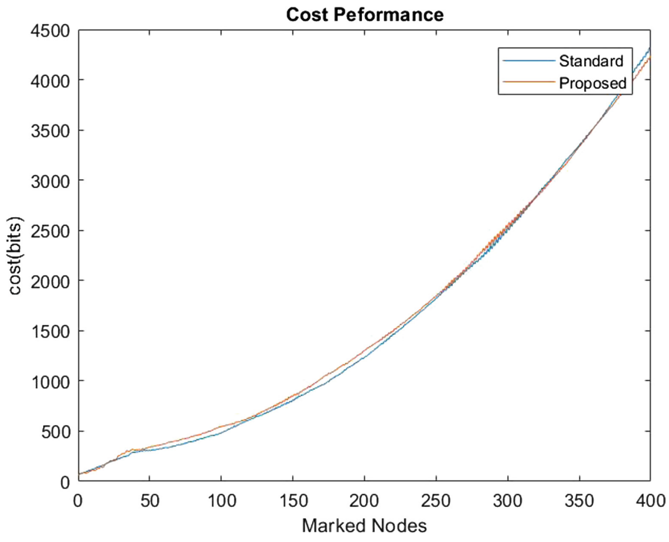


Fig. 8. Cost comparison between standard and proposed algorithm

● Normal Node ● Marked Node ● Newly Marked Node

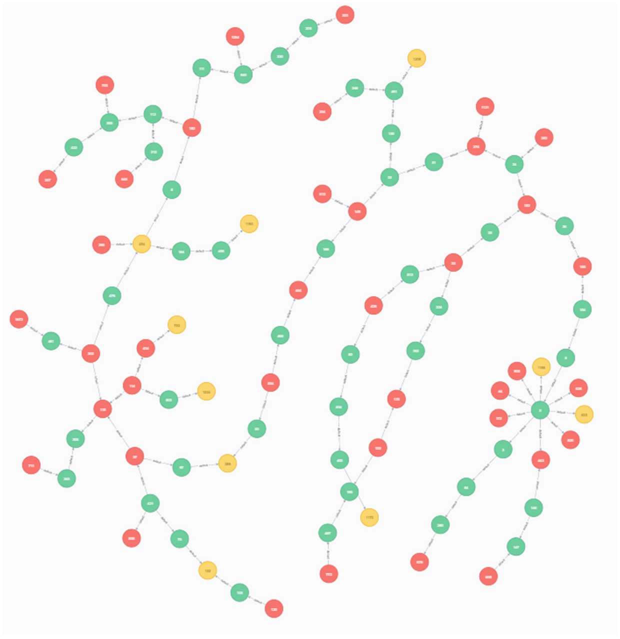


Fig. 9. Dot2Dot algorithm output Partition

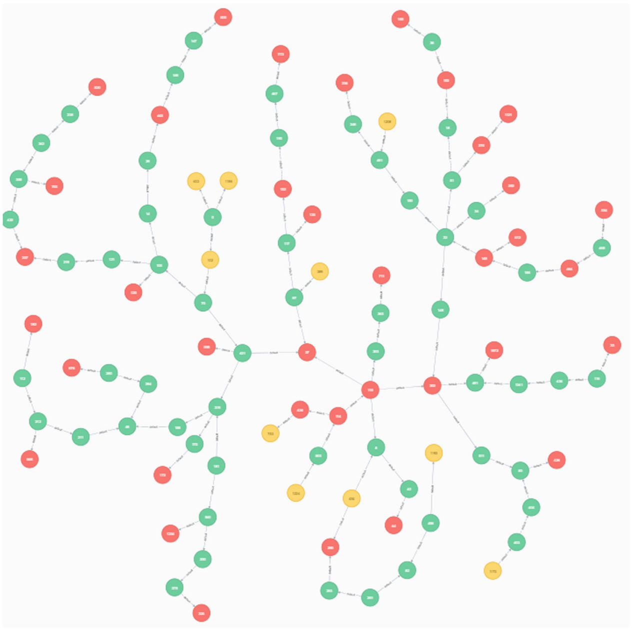


Fig. 10. Proposed algorithm output Partition

4 Conclusion

The proposed implementation drastically reduces the computation time of partitioning input graph into subgraphs by reducing the search space of considered edges of previously marked nodes. Instead of considering all edges originating from a previously marked node, our algorithm considers just one edge. This results in a decrease in time taken from exponential to polynomial time while maintaining the integrity of the graph cost.

This reduction in time allows the algorithm to work seamlessly in an online, real-time setting without compromising on the correctness of output. There are several future applications for this algorithm. Expanding on this technique, the edges could be stored in a Splay Tree to reduce space complexity if the algorithm is to be used in a space-constrained environment. As an example, this could be applied to recommender systems which could quickly find similar users and products. Furthermore, there are applications in identifying anomalous events within a network and grouping the behavior of users within a social network.

References

1. Karypis, G., Kumar, V.: METIS-unstructured graph partitioning and sparse matrix ordering system, version 2.0 (1995)
2. Kwatra, V., Schödl, A., Essa, I., Turk, G., Bobick, A.: Graphcut textures: image and video synthesis using graph cuts. In: ACM Transactions on Graphics (ToG), vol. 22, pp. 277–286. ACM (2003)
3. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. Internet Math. **6**(1), 29–123 (2009)
4. Berkhin, P., et al.: A survey of clustering data mining techniques. In: Kogan, J., Nicholas, C., Teboulle, M. (eds.) Grouping Multidimensional Data, pp. 25–71. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-28349-8_2
5. Ng, R.T., Han, J.: Effcient and effective clustering methods for spatial data mining. In: Proceedings of VLDB, pp. 144–155 (1994)
6. Huang, Z.: A fast clustering algorithm to cluster very large categorical data sets in data mining. DMKD **3**(8), 34–39 (1997)
7. Ding, C.H.Q., He, X., Zha, H., Gu, M., Simon, H.D.: A min-max cut algorithm for graph partitioning and data clustering. In: Proceedings IEEE International Conference on Data Mining, ICDM 2001, pp. 107–114. IEEE (2001)
8. Slota, G.M., Madduri, K., Rajamanickam, S.: Pulp: scalable multi-objective multi-constraint partitioning for small-world networks. In: 2014 IEEE International Conference on Big Data (Big Data), pp. 481–490. IEEE (2014)
9. Kruskal, J.B.: On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. Am. Math. Soc. **7**(1), 48–50 (1956)
10. Prim, R.C.: Shortest connection networks and some generalizations. Bell Labs Tech. J. **36**(6), 1389–1401 (1957)
11. Akoglu, L., Chau, D.H., Vreeken, J., Tatti, N., Tong, H., Faloutsos, C.: Mining connection pathways for marked nodes in large graphs. In: Proceedings of the 2013 SIAM International Conference on Data Mining, pp. 37–45. SIAM (2013)
12. Amazon Co-purchasing Network (2003). http://snap.stanford.edu/data/amazon0302.html. Accessed 02 Mar 2003

A New Global Foreground Modeling and Local Background Modeling Method for Video Analysis

Hang Shi(✉) and Chengjun Liu(✉)

Department of Computer Science, New Jersey Institute of Technology, Newark, NJ 07102, USA
{hs328,cliu}@njit.edu

Abstract. This paper presents a new Global Foreground Modeling (GFM) and Local Background Modeling (LBM) method for video analysis. First, a novel feature vector, which integrates the RGB values, the horizontal and vertical Haar wavelet features, and the temporal difference features of a pixel, enhances the discriminatory power due to its increased dimensionality. Second, the local background modeling process chooses the most significant single Gaussian density to model the background locally for each pixel according to the weights learned for the Gaussian mixture model. Third, an innovative global foreground modeling method, which applies the Bayes decision rule, models the foreground pixels globally. The GFM method thus is able to achieve improved foreground detection accuracy and capable of detecting stopped moving objects. Experimental results using the New Jersey Department of Transportation (NJDOT) traffic video sequences show that the proposed method achieves better video analysis results than the popular background subtraction methods.

Keywords: Global Foreground Modeling (GFM) · Local Background Modeling (LBM) · Background subtraction · Gaussian mixture model · Video analysis · Bayes decision rule

1 Introduction

Video analysis, which has broad applications in surveillance and security, often applies popular techniques in multidisciplinary fields, such as computer vision, pattern recognition, machine learning, and artificial intelligence [1,2]. The representative statistical modeling methods, such as the background subtraction approaches, usually model the background locally, i.e., for each pixel location [3–6]. Local background modeling, which does not apply the foreground knowledge, is suboptimal in achieving the best video analysis performance. Hayman and Eklundh [7] considered both the background and foreground modeling by splitting the Gaussian densities learned in the Gaussian mixture model (GMM)

[6] for the background density and the foreground density, respectively. The foreground modeling using the $K-1$ Gaussian densities from the GMM, however, is still not accurate in the sense of precisely defining the foreground density. Another drawback of the background subtraction approaches is that the foreground objects are quickly absorbed into background when they stop moving, such as the vehicles temporarily stopping in front of traffic lights. As a result, these stopped moving targets are missed in video analysis.

To address these issues, we propose a new Global Foreground Modeling (GFM) method to improve upon the popular background subtraction approaches for video analysis. In particular, we present a new global foreground modeling and Local Background Modeling (LBM) method for video analysis. The novelty of our method includes a new feature vector with enhanced discriminatory power, the local background modeling, the global foreground modeling, and the application of the Bayes classifier for foreground and background classification.

First, feature representation plays a very important role in pattern classification [8–11]. Our recent research shows that the discriminatory power of the feature vector is enhanced by increasing the dimensionality of the feature vector [12]. The popular background subtraction algorithms usually apply the red, green, and blue values of a pixel to define the input vector [1,2]. As a result, the size of the input vector is limited, which restricts the discriminatory power of the vector. Researchers have investigated on this feature representation issue over the years and proposed new methods to address it [13–15]. Varadarajan et al. [13] presented a region based model, which uses small blocks to separate the foreground and the background. The small blocks are able to increase the size of the input vector quadratically in order to increase the discriminative capability [13]. Pandey and Lazebnik [14] proposed deformable part-based models for feature representation and utilized the latent SVM for classification. Wren et al. [15] applied a blob-based feature representation method for real-time detection and tracking of human body. One advantage of the region-based methods is due to the application of the contextual information to enhance the discriminatory power of the input vector. Nevertheless, region based methods may lead to rough outlines and false regions instead of false points, due to the inaccuracy caused by the block level segmentation. We present a new feature vector for feature representation by integrating the RGB values, the horizontal and vertical Haar wavelet features [16], and the temporal difference features. Our new feature vector thus is able to increase the dimensionality of the input pattern vector for each pixel and improve the foreground and background segmentation performance due to its enhanced discriminatory power.

Second, in the novel feature vector space we model the background locally for each pixel according to the weights learned for the Gaussian mixture model. Specifically, we first use the traditional Gaussian mixture model to estimate the probability density function of the pixel at a location [5–7]. We then apply the constant weight updating scheme [17] to learn the parameters of the Gaussian mixture model. And we finally choose the most significant single Gaussian density to model the background locally for each pixel, according to the weights of the Gaussian densities.

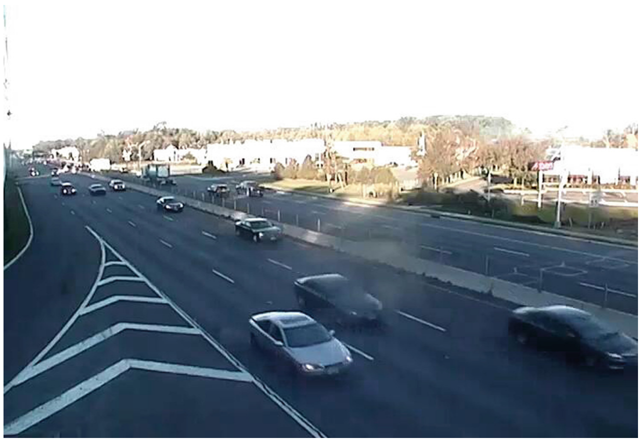


Fig. 1. A frame from a traffic video. The RGB values of some foreground regions, i.e., vehicles, are similar to the RGB values of the background, namely, the lanes of the road.

Third, we present an innovative global foreground modeling method. Our GFM method, which models the foreground pixels globally, is in contrast to the background modeling algorithms where each pixel location is modeled locally. Specifically, by using the Bayes decision rule for minimum error, our GFM method derives one Gaussian distribution for every foreground pixel instead of every location for foreground modeling. In contrast to the local background modeling approach, our GFM method never eliminates a low weight distribution, as every distribution represents a foreground feature vector. The advantages of the GFM method include being able to achieve improved foreground classification performance and being capable of detecting stopped moving objects. Finally, the Bayes classifier is applied to determine whether a pixel belongs to the foreground or the background.

2 A New Feature Vector for Feature Representation

Pixel-based background subtraction algorithms usually apply the red, green, and blue values of a pixel to define the input vector [1,2]. As a result, the size of the input vector is limited, which in turn restricts the discriminatory power of the vector. We propose in this paper a novel feature vector for feature representation by integrating the RGB values, the horizontal and vertical Haar wavelet features [16], and the temporal difference features. Our new feature vector thus is able to increase the dimensionality of the input pattern vector for each pixel and improve the foreground and background segmentation performance due to its enhanced discriminatory power. Next, we briefly explain the elements used to define our feature vector.

First, the new feature vector incorporates the RGB values of a pixel. The RGB values are useful for some simple segmentation tasks as demonstrated in

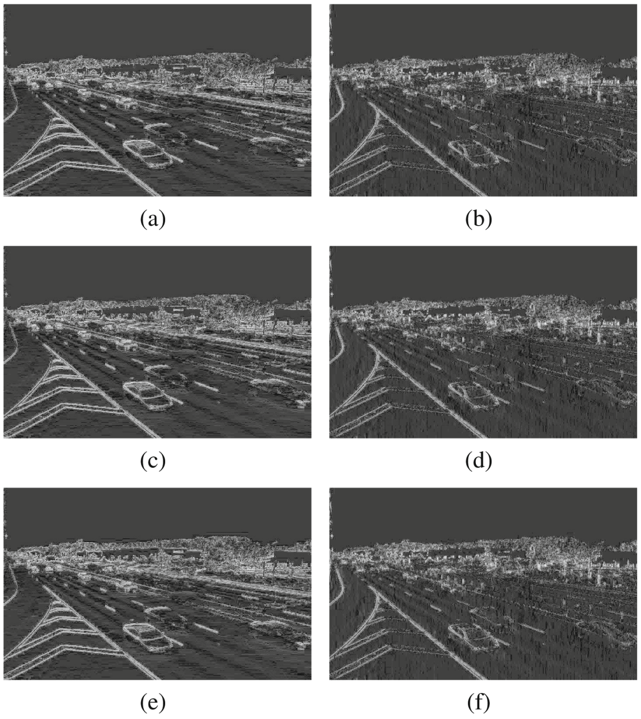


Fig. 2. The first row shows the horizontal and vertical Haar wavelet features of the red component image, respectively. The second and third rows show the horizontal and vertical Haar wavelet features of the green and blue component images, respectively. Note that the figures are displayed in the dynamic range with logarithm scale.

the paper [18]. The RGB values alone, however, are insufficient for achieving good foreground segmentation performance. Figure 1 shows a frame from a traffic video, where the RGB values of some foreground regions, i.e., vehicles, are similar to the RGB values of the background, namely, the lanes of the road. As a result, applying only the RGB values will not satisfactorily segment the foreground from the background. Therefore, our new feature vector will be augmented with additional features.

Second, the new feature vector integrates the horizontal and vertical Haar wavelet features. Haar wavelet features have been broadly applied in computer vision and pattern recognition [16]. Figure 2 shows the horizontal and vertical Haar wavelet features in one frame. Specifically, the first row of Fig. 2 shows the horizontal and vertical Haar wavelet features of the red component image, the second row of Fig. 2 shows the horizontal and vertical Haar wavelet features of the green component image, and the third row of Fig. 2 shows the horizontal and vertical Haar wavelet features of the blue component image.

And finally, the new feature vector combines the temporal difference features. As temporal information plays a crucial role in motion analysis, the temporal

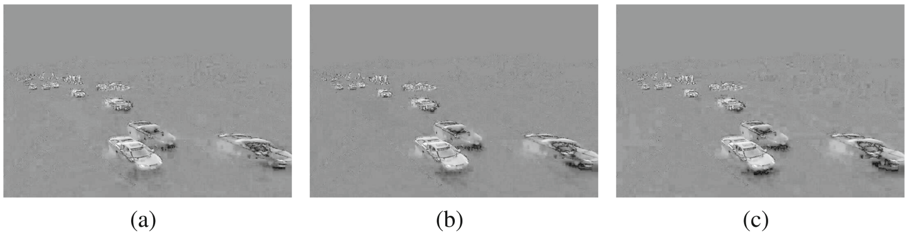


Fig. 3. The temporal difference features for the red, green, and blue component images, respectively. Note that the figures are displayed in the dynamic range with logarithm scale.

difference features are added to our new feature vector. Our idea is to compute the temporal difference between the current frame and the next frame with the goal of distinguishing the moving objects and the stable background. Specifically, the temporal difference is computed for the red, green, and blue component images, respectively. Figure 3 shows the temporal difference features for the red, green, and blue component images, respectively.

To summarize, by integrating the RGB values, the horizontal and vertical Haar wavelet features, and the temporal difference features, our new feature vector resides in a 12 dimensional space with enhanced discriminatory power for foreground or background segmentation.

3 Local Background Modeling Using a Single Gaussian Density

For background modeling, one single Gaussian density is learned in the novel feature vector space $\mathbb{R}^{12}$ to model the background locally for each pixel. Specifically, the local background modeling involves a two-step process. First, the probability density function of the pixel at (i, j) is estimated using the traditional local Gaussian Mixture Model (GMM) [5–7]. The constant weight updating scheme [17] is then applied to learn the parameters of the GMM. Second, according to the weights of the Gaussian densities, we choose the most significant single Gaussian density to model the background locally for each pixel.

The feature vector, $\mathbf{x}_{i,j} \in \mathbb{R}^{12}$, is defined by the RGB values of the pixel, the horizontal and vertical Haar wavelet features, and the temporal difference features. For notational simplicity and without loss of generality, we will drop the subscripts i, j in the following equations. The probability density function of the pixel at (i, j) may be estimated as follows [6]:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \alpha_k N(\mathbf{M}_k, \Sigma_k) \tag{1}$$

where $N(\mathbf{M}_k, \Sigma_k) = \frac{1}{(2\pi)^{d/2}|\Sigma_k|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \mathbf{M}_k)^t \Sigma_k^{-1}(\mathbf{x} - \mathbf{M}_k)\right\}, \sum_{k=1}^{K} \alpha_k = 1$, K indicates the number of Gaussian densities in the GMM model,

α_k is the weight for each Gaussian density, d is the dimensionality of the feature vector $\mathbf{x}$, and $\mathbf{M}_k$ is the mean vector and Σ_k is the covariance matrix for the k-th Gaussian density.

For simplicity, we assume that the covariance matrix Σ_k is diagonal. Note that the traditional pixel-based background subtraction algorithms use the same assumption as well [5–7]. Generally speaking, the GMM is a comprehensive model for describing complex scenes with various activities. Thus both the background and the various activities are described by the different Gaussian densities. Note that the background, which is usually static without many changes, may be modeled by one Gaussian density with a large weight.

We use the constant weight updating scheme [17] to train the GMM, namely, to learn the parameters specified in Eq. 1. The initialization process first sets the mean vector, the covariance matrix, and the weight for each Gaussian density of the pixel equal to zero. We then define a counter $n_k, k = 1, \cdots, K$ for each Gaussian density to count the number of pattern vectors that satisfy Eq. 2. Note that the $n'_k s$ are initialized to zero. For each Gaussian density the following criterion is evaluated:

$$(\mathbf{x} - \mathbf{M}_k)^t(\mathbf{x} - \mathbf{M}_k) < \prod_{i=1}^{K} \sigma_i^2 \tag{2}$$

where $\sigma_i^2, i = 1, \cdots, K$, are the diagonal elements of Σ_k, which is a diagonal matrix. If the k-th Gaussian density satisfies this condition, we update the parameters of the Gaussian density as follows:

$$\mathbf{M}'_k = (n_k\mathbf{M}_k + \mathbf{x})/(n_k + 1) \tag{3}$$
$$\Sigma'_k = (n_k\Sigma_k + (\mathbf{x} - \mathbf{M}_k)(\mathbf{x} - \mathbf{M}_k)^t)/(n_k + 1) \tag{4}$$
$$n'_k = n_k + 1 \tag{5}$$

If no existing Gaussian density satisfies Eq. 2, then the pattern vector defines a new Gaussian density function, whose parameters are determined as follows: the pattern vector is the mean vector, and a predefined value multiplied by the identity matrix is the covariance matrix. The counter is set equal to one. In the GMM model, if there exists a Gaussian density with a zero weight, then the new Gaussian density replaces it. If not, the new Gaussian density replaces the existing Gaussian density function with the smallest weight.

We further update the weights of all the Gaussian densities as follows:

$$\alpha'_k = n'_k / \sum_{k=1}^{K} n'_k \tag{6}$$

After updating the weights, we sort all the Gaussian densities in the descending order.

Note that the background is usually static without many changes. Therefore, it may be modeled by one Gaussian density with the largest weight. We thus choose the most significant Gaussian density, which is the first density in the GMM, to model the background.

$$p(\mathbf{x}|\omega_b) = N\ (\mathbf{M}'_1, \Sigma'_1) \tag{7}$$

where ω_b represents the background class.

4 Novel Global Foreground Modeling

We present in this section a novel Global Foreground Modeling (GFM) method that models every foreground pixel as a Gaussian distribution. In contrast to the background modeling algorithms where each location (i, j) is modeled as a mixture of Gaussian distribution, our GFM method, which does not model any specific location, is global in nature. Furthermore, for every foreground pixel, our GFM method derives one Gaussian distribution for foreground modeling using the Bayes decision rule for minimum error.

In particular, the GFM method learns M Gaussian densities for defining the M conditional density functions for the foreground pixels, respectively: $p(\mathbf{x}|\omega_1), p(\mathbf{x}|\omega_2), \cdots, p(\mathbf{x}|\omega_M)$. For initialization, we set all the mean vectors and the covariance matrices of the M Gaussian densities equal to zero, and set the counter of the number of the input feature vectors and the weight for each Gaussian density equal to zero. After initialization, for a new input feature vector, we first check whether the GFM model has a Gaussian density with zero weight. If yes, we check using Eq. 2 to see if the input feature vector can be absorbed by the background Gaussian density function. If it can be absorbed by the background Gaussian density function, we continue without modifying the GFM model. If it cannot be absorbed by the background Gaussian density function or by the non-zero weighted GFM Gaussian densities, a new Gaussian density function is created with the input feature vector as the mean vector and a predefined value multiplied by the identity matrix as the covariance matrix. The counter is set equal to one.

If none of the GFM Gaussian densities have zero weights, we then apply the Bayes decision rule for minimum error to identify one Gaussian density for the input feature vector. As a result, the Gaussian density becomes the conditional probability density function for the foreground. Specifically, the conditional probability density function for the foreground is derived by means of the Bayes decision rule for minimum error [19]:

$$p(\mathbf{x}|\omega_f)P(\omega_f) = \max_{i=1}^{M}\{p(\mathbf{x}|\omega_i)P(\omega_i)\} \tag{8}$$

Thus, by using the Bayes decision rule for minimum error, we can classify the input feature vector $\mathbf{x}$ into one Gaussian distribution of the GFM method: $p(\mathbf{x}|\omega_f)$. The $p(\mathbf{x}|\omega_f)$ will be used as the foreground conditional probability density function in the next segmentation step. Note that $\mathbf{x}$ is further used to update that Gaussian density function and the weights in our GFM model, and the updating process of the mean vector, the covariance matrix, and the weight

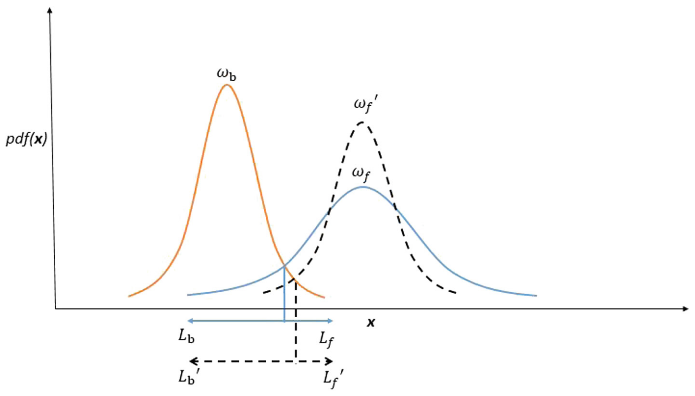


Fig. 4. The Bayes error is reduced if the variances for the conditional probability density function for the foreground decrease. L_f and L_b represent regions, and L'_f and L'_b denote new regions.

values in the GFM model is similar to what we discussed before in the previous section. The difference is we never eliminate a low weight distribution. In the GMM model, a low weight component stands for a feature vector that rarely appears at one location. As a result, such a component could be eliminated because it has a low possibility to be a background pixel. But in the GFM model, every distribution stands for a foreground feature vector. We cannot ignore any possibility of a presented value. So the GFM will absorb every foreground pixel into one distribution, even if by doing so, it will increase the variance of a distribution.

Our GFM method has the following advantages. First, the GFM method is able to achieve improved classification performance due to the application of the Bayes decision rule for minimum error. To understand this advantage, we now briefly review the Bayes error. The conditional error given **x** is defined as follows:

$$r(e|\mathbf{x}) = min[P(\omega_b|\mathbf{x}), P(\omega_f|\mathbf{x})] \tag{9}$$

The Bayes error, which is the expectation of the conditional error, is defined as follows:

$$\epsilon = \int r(e|\mathbf{x})p(\mathbf{x})d\mathbf{x} = P(\omega_b)\int_{L_f} p(\mathbf{x}|\omega_b)d\mathbf{x} + P(\omega_f)\int_{L_b} p(\mathbf{x}|\omega_f)d\mathbf{x} \tag{10}$$

where L_f and L_b are the regions shown in Fig. 4. We can see from Fig. 4 that if we are able to define the conditional probability density function for the foreground with smaller variances, the Bayes error could be reduced—see the L'_f and L'_b regions in Fig. 4.

We next demonstrate that our GFM method is able to achieve improved classification performance. Note that the GFM method applies the Bayes decision rule for minimum error to derive the conditional probability density function

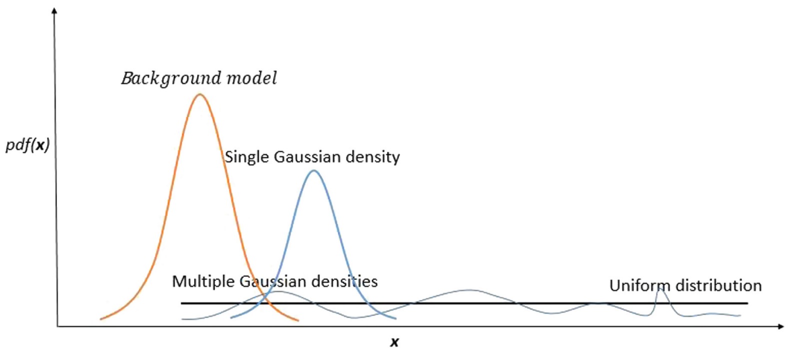


Fig. 5. The GFM's single Gaussian density for foreground modeling yields smaller Bayes error when compared with the multiple Gaussian densities or the uniform distribution for foreground modeling.

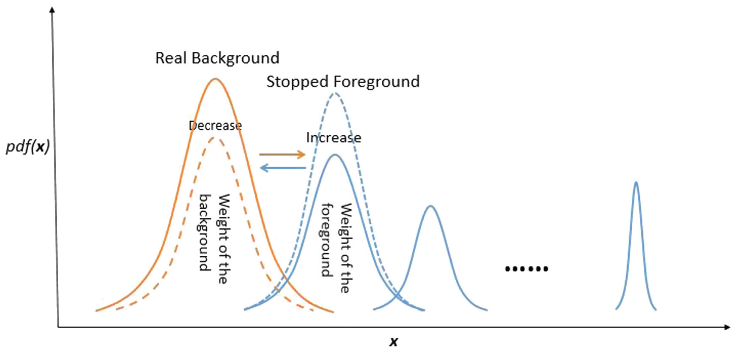


Fig. 6. When a foreground object stops moving, the weight of the Gaussian density for the background pixel will decrease, and the weight of the Gaussian density for the stopped foreground pixel will increase.

for the foreground. Other popular methods model the foreground differently. Zivkovic [5] assumed that the foreground has a uniform distribution. Hayman and Eklundh [7] used $K-1$ Gaussian densities to model the foreground. Figure 5 shows that our GFM's single Gaussian density for foreground modeling tends to yield smaller Bayes error when compared with the multiple Gaussian densities or the uniform distribution for foreground modeling. Therefore, our GFM method, which derives one Gaussian distribution for foreground modeling using the Bayes decision rule for minimum error, is able to achieve improved classification performance.

Second, the GFM is able to detect the stopped moving objects. Most of the popular background subtraction methods have difficulty detecting the foreground objects when they stop moving, and these foreground objects are quickly classified as the background [1,2]. The reason for such a misclassification is due to the Gaussian mixture modeling applied by the popular background subtrac-

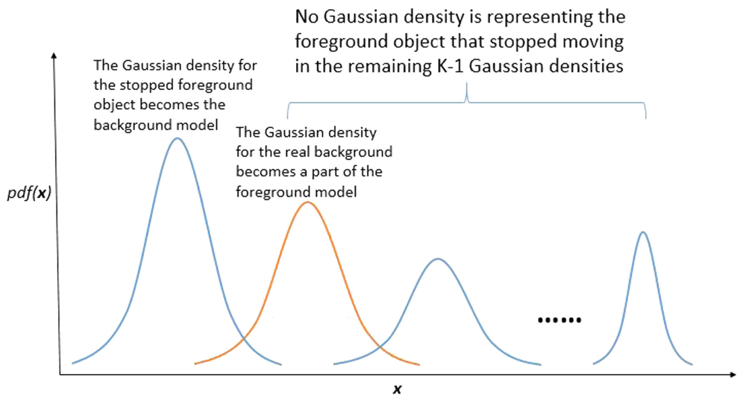

No Gaussian density is representing the foreground object that stopped moving in the remaining K-1 Gaussian densities
The Gaussian density for the stopped foreground object becomes the background model
The Gaussian density for the real background becomes a part of the foreground model
pdf(x)
x
......

Fig. 7. After the Gaussian density for the foreground becomes the background model, the foreground model, which is defined by the remaining $K - 1$ Gaussian densities, will no longer have a Gaussian density representing the foreground object that stopped moving.

tion methods [5,6]—see Eq. 1. Specifically, Fig. 6 shows that when a foreground object stops moving, the weight of the Gaussian density for the background pixel will decrease, and the weight of the Gaussian density for the stopped foreground pixel will increase. When the foreground object stops a little longer, according to Eq. 1, the Gaussian density for the foreground pixel becomes the background model, and as a result, the foreground objects become the background and are no longer detected. Even though Hayman and Eklundh's method [7] applies both the background and foreground modeling by using the $K-1$ Gaussian densities to model the foreground. It still does not model the foreground objects when they stop moving. In particular, Fig. 7 shows that after the Gaussian density for the foreground becomes the background model, the foreground model, which is defined by the remaining $K - 1$ Gaussian densities, will no longer have a Gaussian density representing the foreground object that stopped moving.

Our GFM method, however, is able to detect the stopped moving objects, such as the vehicles that stop in front of the traffic lights. The reason is because of the global modeling of the foreground pixels, which is in contrast to the location based background modeling. In essence, our GFM method chooses globally the Gaussian density according to the Bayes decision rule for minimum error to model the foreground pixel as shown in Eq. 8. As a result, when an object stops moving, the foreground model will still maintain the accurate Gaussian Density to model it. The final classification of foreground pixels depends on the Bayes classifier that will be explained in the next section.

5 Foreground and Background Classification

The Bayes classifier is applied to determine whether a pixel belongs to the foreground or the background. The conditional probability density functions for the foreground and the background, $p(\mathbf{x}|\omega_f)$, $p(\mathbf{x}|\omega_b)$, are defined in the previous sections, respectively. As discussed in Sect. 3, the first Gaussian density is chosen as the background model, the prior probability for the background, $P(\omega_b)$, is estimated using the weight for this Gaussian density: $P(\omega_b) = \alpha_1$. The prior probability for the foreground, $P(\omega_f)$, is then estimated as follows: $P(\omega_f) = 1 - \alpha_1$.

Given a pixel, its feature vector, $\mathbf{x} \in \mathbb{R}^{12}$, is first defined by the RGB values of the pixel, the horizontal and vertical Haar wavelet features, and the temporal difference features. The feature vector, $\mathbf{x}$, is then classified to either the foreground class or the background class according to the following discriminant function:

$$h(\mathbf{x}) = p(\mathbf{x}|\omega_f)P(\omega_f) - p(\mathbf{x}|\omega_b)P(\omega_b) \tag{11}$$

Namely, the feature vector $\mathbf{x}$ is classified to the foreground class if $h(\mathbf{x}) > 0$, and to the background class otherwise.

6 Experiments

We use the New Jersey Department of Transportation (NJDOT) traffic video sequences to evaluate our proposed method and compare it with the popular background subtraction algorithms, such as the Gaussian Mixture Model (GMM) [5–7]. Specifically, we first evaluate the computational efficiency of our proposed Global Foreground Modeling and Local Background Modeling (GFM-LBM) method to demonstrate its efficiency for real time implementation. We then comparatively evaluate our GFM-LBM method and the popular GMM method to reveal the improved foreground segmentation accuracy by our GFM-LBM method. We finally carry out experiments to show that our GFM-LBM method is able to detect the foreground objects that stopped moving, while in comparison the popular GMM method fails to detect such foreground objects.

We now evaluate the computational efficiency of the GFM-LBM method using the NJDOT traffic video sequences. In particular, there are three types of spatial resolutions of the NJDOT videos: the existing quality videos with the spatial resolution of 352×240, the enhanced quality videos with the spatial resolution of 704×480, and the highest quality videos with the spatial resolution of 752×480. The frame rates for these three types of videos are 15 fps, 15 fps, and 30 fps, respectively. The computer we use is a DELL XPS 8900 PC with a 3.4 GHz processor and 16 GB RAM. The parameters for the GFM-LBM method are as follows: the number of Gaussian density functions for the GMM is 3, and the number of Gaussian density functions for the GFM is 5. Table 1 shows the experimental results using the three types of the NJDOT traffic videos: 352×240 video, 704×480 video, and 752×480 video. Specifically, the run time for the 752×480 video is 82 ms per frame, the run time for the 704×480 video is 68 ms per frame, and the run time for the 352×240 video is 15 ms per frame. These experimental results indicate that the GFM-LBM method is able to process the existing quality videos and the enhanced quality videos in real time on the DELL XPS 8900 PC with a 3.4 GHz processor. As a matter of fact, the existing quality

Table 1. The run time of the GFM-LBM method using the three types of the NJDOT traffic videos: 352 × 240 video, 704 × 480 video, and 752 × 480 video.

Video	752 × 480 video	704 × 480 video	352 × 240 video
Run time	82 ms/frame	68 ms/frame	15 ms/frame

videos are currently in use by NJDOT for traffic monitoring. The other two types are collected using temporary cameras at some experiment zones.

We next comparatively evaluate our GFM-LBM method and the popular GMM method to show the improved foreground segmentation accuracy and robustness by our GFM-LBM method. Towards that end, Fig. 8 shows the comparative foreground detection performance of the GFM-LBM method and the popular GMM method. In particular, Fig. 8(a) displays an original frame from an NJDOT video, and Fig. 8(b) shows the ground truth of foreground detection. Figure 8(c) shows the foreground detection results using the GFM-LBM method, and in comparison Fig. 8(d) shows the foreground detection results using the

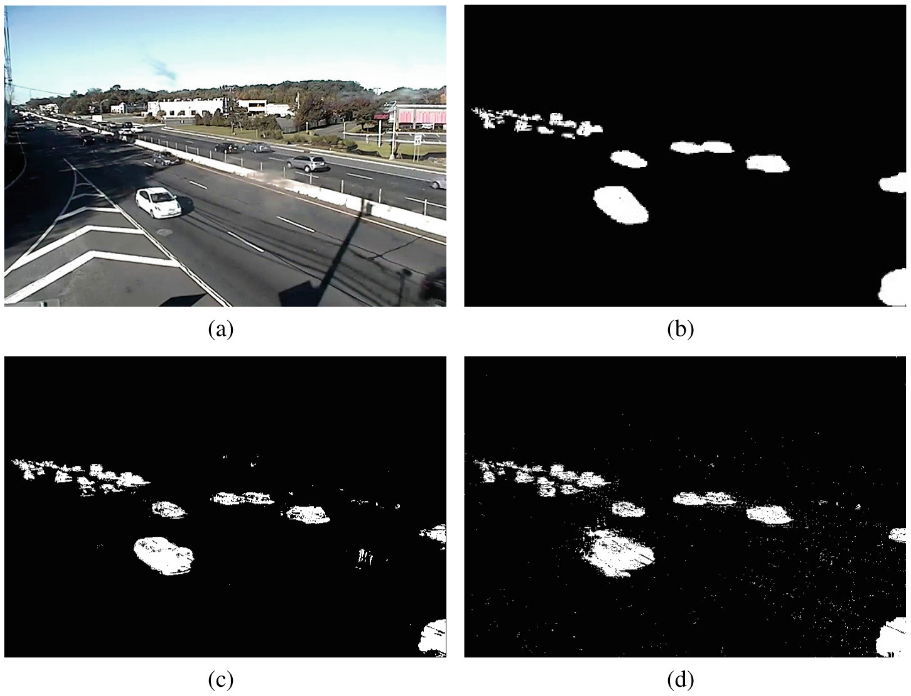


Fig. 8. Comparative foreground detection performance of the GFM-LBM method and the popular GMM method. (a) An original frame. (b) Ground truth of foreground detection. (c) Foreground detection by the GFM-LBM method. (d) Foreground detection by the popular GMM method.

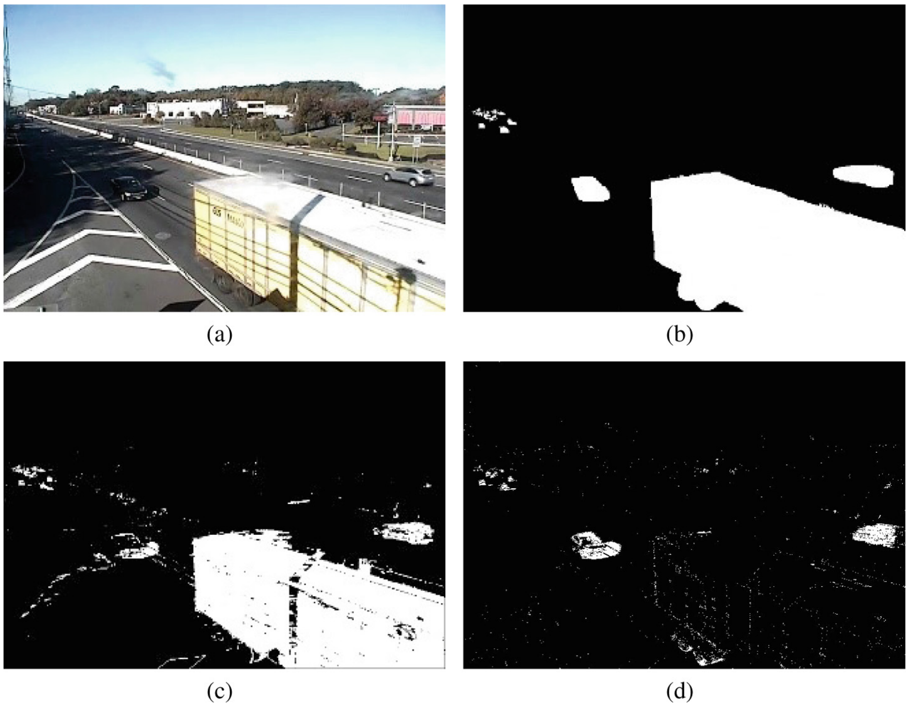

(a) (b) (c) (d)

Fig. 9. Comparative foreground detection performance of the GFM-LBM method and the popular GMM method. (a) An original frame from an NJDOT video. (b) Ground truth of foreground detection. (c) Foreground detection results using the GFM-LBM method. (d) Foreground detection results using the popular GMM method.

popular GMM method. Figure 8(c) and (d) reveal that the GFM-LBM method achieves better foreground detection results than the traditional GMM method in terms of better edges and fewer false positive pixels.

We finally implement experiments to show that our GFM-LBM method is able to detect the foreground objects that stopped moving, while in comparison the popular GMM method fails to detect such foreground objects. In traffic monitoring vehicles often stop in congestion or in front of traffic lights, but these vehicles are still the foreground objects and should be detected for traffic video analysis. The traditional GMM method, however, quickly absorbs these foreground objects into background when they stop moving, which leads to incorrect foreground segmentation. Figure 9 shows the comparative foreground segmentation results using the GFM-LBM method and the popular GMM method using the NJDOT videos. Specifically, Fig. 9(a) shows an original frame from an NJDOT video, and Fig. 9(b) shows the ground truth of foreground detection. Figure 9(c) shows the foreground detection results using the GFM-LBM method, and in comparison Fig. 9(d) shows the foreground detection results using the popular GMM method. Figure 9(c) and (d) show that the GFM-LBM method is able to detect the vehicles even when they stop moving, but in comparison the popular GMM method fails to detect these foreground vehicles.

7 Conclusions

We have presented in this paper a new Global Foreground Modeling (GFM) and Local Background Modeling (LBM) method for video analysis. The major contributions of the paper are three aspects. First, a novel feature vector is proposed by integrating the RGB values, the horizontal and vertical Haar wavelet features, and the temporal difference features of a pixel for feature representation. The new feature vector, which enhances the discriminatory power due to its increased dimensionality, is able to improve the foreground and background segmentation performance for video analysis. Second, the local background modeling process, which capitalizes on the traditional Gaussian mixture models, is explained by choosing the most significant single Gaussian density to model the background locally for each pixel according to the weights learned for the Gaussian mixture model. Third, an innovative global foreground modeling or GFM method, which models the foreground pixels globally, is presented for foreground modeling. The GFM method, which applies the Bayes decision rule for minimum error, derives one Gaussian distribution for every foreground pixel. The advantages of the GFM method include being able to achieve improved foreground classification performance and capable of detecting stopped moving objects. Finally, the Bayes decision rule for minimum error is applied to determine whether a pixel belongs to the foreground or the background. Experimental results using the New Jersey Department of Transportation (NJDOT) traffic video sequences have shown that the proposed method achieves better video analysis results than the popular background subtraction methods.

Acknowledgments. This paper is partially supported by the NSF grant 1647170.

References

1. Bouwmans, T., Sobral, A., Javed, S., Jung, S.K., Zahzah, E.: Decomposition into low-rank plus additive matrices for background/foreground separation: a review for a comparative evaluation with a large-scale dataset. Comput. Sci. Rev. **23**, 1–71 (2017)
2. Sobral, A., Vacavant, A.: A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. Comput. Vis. Image Underst. **122**, 4–21 (2014)
3. Bouwmans, T., Silva, C., Marghes, C., Zitouni, M.S., Bhaskar, H., Frélicot, C.: On the role and the importance of features for background modeling and foreground detection. CoRR abs/1611.09099 (2016)
4. Barnich, O., Droogenbroeck, M.V.: ViBe: a universal background subtraction algorithm for video sequences. IEEE Trans. Image Process. **20**(6), 1709–1724 (2011)
5. Zivkovic, Z.: Improved adaptive Gaussian mixture model for background subtraction. In: 17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, 23–26 August 2004, pp. 28–31. IEEE Computer Society (2004)

6. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: 1999 Conference on Computer Vision and Pattern Recognition (CVPR 1999), 23–25 June 1999, Ft. Collins, CO, USA, pp. 2246–2252. IEEE Computer Society (1999)
7. Hayman, E., Eklundh, J.: Statistical background subtraction for a mobile observer. In: 9th IEEE International Conference on Computer Vision (ICCV 2003), 14–17 October 2003, Nice, France, pp. 67–74. IEEE Computer Society (2003)
8. Liu, C. (ed.): Recent Advances in Intelligent Image Search and Video Retrieval. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-319-52081-0
9. Liu, Q., Liu, C.: A novel locally linear KNN method with applications to visual recognition. IEEE Trans. Neural Netw. Learn. Syst. **28**(9), 2010–2021 (2017)
10. Puthenputhussery, A., Liu, Q., Liu, C.: A sparse representation model using the complete marginal fisher analysis framework and its applications to visual recognition. IEEE Trans. Multimed. **19**(8), 1757–1770 (2017)
11. Chen, S., Liu, C.: Eye detection using discriminatory haar features and a new efficient SVM. Image Vis. Comput. **33**, 68–77 (2015)
12. Chen, S., Liu, C.: Clustering-based discriminant analysis for eye detection. IEEE Trans. Image Process. **23**(4), 1629–1638 (2014)
13. Varadarajan, S., Miller, P.C., Zhou, H.: Region-based mixture of gaussians modelling for foreground detection in dynamic scenes. Pattern Recogn. **48**(11), 3488–3503 (2015)
14. Pandey, M., Lazebnik, S.: Scene recognition and weakly supervised object localization with deformable part-based models. In: Metaxas, D.N., Quan, L., Sanfeliu, A., Gool, L.J.V. (eds.): IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6–13, 2011, pp. 1307–1314. IEEE Computer Society (2011)
15. Wren, C.R., Azarbayejani, A., Darrell, T., Pentland, A.: Pfinder: real-time tracking of the human body. IEEE Trans. Pattern Anal. Mach. Intell. **19**(7), 780–785 (1997)
16. Liu, C.: A Bayesian discriminating features method for face detection. IEEE Trans. Pattern Anal. Mach. Intell. **25**(6), 725–740 (2003)
17. Mittal, A., Huttenlocher, D.P.: Scene modeling for wide area surveillance and image synthesis. In: 2000 Conference on Computer Vision and Pattern Recognition (CVPR 2000), 13–15 June 2000, Hilton Head, SC, USA, pp. 2160–2167. IEEE Computer Society (2000)
18. Grove, T., Baker, K.D., Tan, T.N.: Colour based object tracking. In: Jain, A.K., Venkatesh, S., Lovell, B.C. (eds.) Fourteenth International Conference on Pattern Recognition, ICPR 1998, Brisbane, Australia, 16–20 August, 1998, pp. 1442–1444. IEEE Computer Society (1998)
19. Webb, A.R.: Statistical Pattern Recognition. Wiley, Hoboken (2003)

A Deep Learning Based Automatic Severity Detector for Diabetic Retinopathy

Rawan AlSaad[1,2](✉), Somaya Al-maadeed[2], Md. Abdullah Al Mamun[2], and Sabri Boughorbel[1]

[1] Systems Biology Department, Sidra Medicine, Doha, Qatar
ralsaad@sidra.org

[2] Department of Computer Science and Engineering, Qatar University, Doha, Qatar

Abstract. Automated Diabetic Retinopathy (DR) screening methods with high accuracy have the strong potential to assist doctors in evaluating more patients and quickly routing those who need help to a specialist. In this work, we used Deep Convolutional Neural Network architecture to diagnosing DR from digital fundus images and accurately classifying its severity. We train this network using a graphics processor unit (GPU) on the publicly available Kaggle dataset. We used Theano, Lasagne, and cuDNN libraries on two Amazon EC2 p2.xlarge instances and demonstrated impressive results, particularly for a high-level classification task. On the dataset of 30,262 training images and 4864 testing images, our model achieves an accuracy of 72%. Our experimental results showed that increasing the batch size does not necessarily speed up the convergence of the gradient computations. Also, it demonstrated that the number and size of fully connected layers do not have a significant impact on the performance of the model.

Keywords: Deep learning · Convolutional Neural Networks
Medical imaging · Diabetic retinopathy

1 Introduction

Diabetic Retinopathy is damage to the retina caused by complications of diabetes which can lead to blindness. It occurs due to changes in the blood vessels in the retina [1]. Sometimes these vessels swell and leak fluid or even close off completely. In other cases, abnormal new blood vessels grow on the surface of the retina. Diabetic retinopathy is the fastest growing cause of blindness, with nearly 415 million diabetic patients at risk worldwide [2]. If caught early, the disease can be treated; if not, it can lead to irreversible blindness. Unfortunately, medical specialists capable of detecting the disease are not available in many parts of the world where diabetes is prevalent. Machine learning (ML) has been leveraged for a variety of classification tasks including automated classification of diabetic retinopathy. We believe that ML can help doctors identify patients in need, particularly among underserved populations. However, much of the previous work which applied ML for automatic DR has focused on "feature-engineering", which involves computing explicit features specified by experts, resulting in algorithms designed to detect specific lesions or predicting the presence of any level of diabetic retinopathy. Deep learning is a machine learning technique that avoids such engineering and allows an algorithm to program itself by learning the most predictive features directly from the images given a large data set of labeled examples, removing the need to specify rules explicitly. Automated DR screening methods with high accuracy have the strong potential to assist doctors in evaluating more patients and quickly routing those who need help to a specialist.

Each eye of the patient can be in one of the 5 levels: from 0 to 4, where 0 corresponds to the healthy state and 4 is the most severe state. Different eyes of the same person can be at different levels (although some previous research managed to leverage the fact that two eyes are not completely independent). The problem is to design and implement a model which can determine the presence of retinopathy and classify its level automatically, without doctors assistance.

Worldwide diabetic retinopathy rates have significantly increased during the past few years. The European Society of Retina Specialists (EURETINA) Congress, held in Germany in late 2013, has called for diabetic retinopathy to be classified as an epidemic. As of 2015, 59.8 million diabetic patients are at risk of diabetic retinopathy in the Middle East and North Africa, [2]. Diabetes is also a major concern to the population in Qatar, affecting about 17% of the population. In addition, diabetic retinopathy is the most common chronic complication of diabetes, affecting about 40% of diabetic patients in Qatar [3,4]. Automated DR screening methods with high accuracy have the strong potential to assist doctors in evaluating more patients and quickly routing those who need help to a specialist. The need for a comprehensive and automated method of DR screening has long been recognized, and previous efforts have made good progress using image classification, pattern recognition, and machine learning.

In this work, we will study deep learning techniques described in recent literature and combine it with our own ideas to realize an, as good as possible, deep learning model for the detection of diabetic retinopathy in retina images. More specifically, we will evaluate the performance of different Deep Convolutional Neural Networks (ConvNets) architectures and the impact of changing its hyperparameters. We used Deep Convolutional Neural Networks (ConvNets) trained on graphical processing units (GPUs) to train the publicly available dataset provided by Kaggle [5]. We used Theano, Lasagne, and cuDNN libraries on two Amazon EC2 p2.xlarge instances. On the dataset of 30,262 training images and 4864 testing images, our model achieved a Kappa of 0.72. Our experimental results showed that increasing the batch size does not necessarily speed up the convergence of the gradient computations. In addition, it demonstrated that the number and size of fully connected layers does not have a significant impact on the model performance.

2 Related Works

While other forms of machine learning have been used to diagnose diabetic retinopathy in the past (such as SVM and neural networks), deep learning is a more pure form of artificial intelligence in that it doesn't receive any guidance to look for particular features. Instead, it learns on its own from nothing but the images and information about what's in them. In this section, we discuss some examples from recent literature which have used deep learning for diabetic retinopathy detection and classification. Gulshan et al. [6] have applied deep learning to create an algorithm for automated detection of diabetic retinopathy and diabetic macular edema in retinal fundus photographs. They trained a deep ConvNet using a retrospective development dataset of 128,175 retinal images. The resultant algorithm was validated in 2016 using two separate dataset EyePACS-1 (9963 images) and Messidor-2 (1748 images). For EyePACS-1, the sensitivity was 90.3%, and the specificity was 98.1%. For Messidor-2, the sensitivity was 87.0%, and the specificity was 98.5%. Further research is necessary to determine the feasibility of applying this algorithm in the clinical setting and to determine whether the use of the algorithm could lead to improved care and outcomes compared with the current ophthalmologic assessment.

Haloi [7] proposed a new deep learning method for microaneurysm detection. Compared to other deep neural networks, it required less preprocessing, vessel extraction and more deep layers for training and testing the fundus image dataset. It consists of five layers which include convolutional, max pooling and Softmax layer with additional dropout training for improving accuracy. A performance of 96% accuracy with 96% specificity and 97% sensitivity was achieved. Wang et al. [8] used ConvNets to perform as a trainable hierarchical feature extractor and Random Forest (RF) as a trainable classifier. It has six stacked layers of convolution and followed by subsampling layers for feature extraction. Random Forest algorithm is utilized for classifier ensemble method and introduced in the retinal blood vessel segmentation. By integrating the merits of feature learning and traditional classifier, the proposed method can automatically learn features from the raw images and predict the patterns. This architecture was tested with two public retinal images databases DRIVE and STARE. It achieved an accuracy of 98% with 82% sensitivity and 97% specificity for the DRIVE database images, and an accuracy of 98% with 81% sensitivity and 98% specificity for the STARE database images.

Melinscak et al. [9] proposed an automatic segmentation of blood vessels in fundus images using deep neural networks. It contains a deep max-pooling ConvNets to segment blood vessels. It deployed 10-layer architecture for achieving a maximum accuracy but worked with small image patches. It includes a preprocessing for resizing and reshaping the fundus images. It carried around 4-convolutional and 4-max pooling layer with two additional fully connected layers for vessel segmentation. This method achieved an average accuracy of 94.7% and an average AUC of 97.5%. The results were tested on publicly available dataset DRIVE. Fraz et al. [10] conducted a survey about the retinal vessel segmentation algorithms. The performance of algorithms was analyzed on two publicly available databases DRIVE and STARE using a number of measures which include accuracy, true positive rate, false positive rate, specificity and area under the receiver operating characteristic (ROC) curve. The algorithms achieved average accuracy in the range of 87.7% to 96% and AUC from 89.84% to 96.1%.

Pratt et al. [11] proposed a ConvNet approach to diagnosing DR from digital fundus images and accurately classifying its severity. They developed a network with CNN architecture and data augmentation which can identify the intricate features involved in the classification task such as microaneurysms, exudate, and hemorrhages on the retina, and consequently provide an automatic diagnosis without user input. The network was trained using high-end GPU on the publicly available Kaggle dataset, which included 80,000 digital fundus images. The model demonstrated magnificent results. It achieved an accuracy of 75% and sensitivity of 95% on 5,000 validation images. In addition, many contestants in the Kaggle DR competition have published their solutions. Most of the solutions used ConvNet architectures implemented using Keras, Caffe, and Theano frameworks for deep learning.

3 Approach

3.1 The Dataset, Hardware, Software

In this project, we will use retina images from the Kaggle [5] competition Diabetic Retinopathy Detection of 2015. The dataset we used contains 35,126 JPEG images (32.5 GB) and a CSV file where the level of the disease (label) is provided for all images. We used 86% of the dataset (30,262 images) for training and 14% (4864 images) for testing. Left eye and right eye images are provided for every subject and images are named using the patient id as well as either left or right (e.g. 1_left.jpeg is the left eye of patient id 1). The images are taken using different types of equipment with the varying field of view, blurring, contrast, color spectrum, and sizes of images. Given the images for which a clinician has rated the presence of diabetic retinopathy on a scale from 0 to 4 (0: No DR, 1: Mild, 2: Moderate, 3: Severe, 4: Proliferative DR), the aim of the competition was to provide a model which can determine the presence of retinopathy automatically, without doctors assistance. The distribution of images of different levels in the

Table 1. Distribution of images of different levels in the dataset

Levels	Number of images	Percentage
0	25810	73.48%
1	2443	73.48%
2	5292	73.48%
3	873	73.48%
4	708	73.48%

dataset set was very uneven, as more than half of the images were of good eyes as illustrated in Table 1.

We used the same software setup used in [12]. We used the following frameworks and libraries: Theano [13], Lasagne [14], python, pylearn2 [15], and CUDA [16]. We used two Amazon EC2 p2.xlarge instances, which are robust, scalable instances that provide GPU-based parallel compute capabilities. The number of parallel processing cores is 2496 with 61 GiB memory.

3.2 Computational Considerations

Since modern GPUs has low memory, we should consider the largest bottleneck of the given memory when building a ConvNet architecture. Sources to keep track of memory are as follows: firstly, from the average volume size which is a raw number of activation at every layer of the ConvNet, and their gradients of equal size. Practically, the activation values are kept on the earlier layers for backpropagation. Only in a testing phase, the current activation value can be kept in any layer by discarding the previous value of the following layers which is an example of a smart implementation. Secondly, it can be known from the parameter size which is a number that holds the network parameters, their gradients during backpropagation. Hence, the memory to be stored the parameter vector must be multiplied by a factor of three at least. Finally, every ConvNet implementation should maintain different memory, for example, the image data batches augmented versions. When a rough estimate of the total number of values is ready for activations, gradients, and miscellaneous (the number should be converted to size in GB). To get the raw number in a byte, multiply the value by 4 (since every floating point is 4 bytes, or maybe by 8 for double precision), and then divide by 1024 to get the amount in KB, MB, and finally GB. Since the activations consume most of the memory, a common heuristic make it fit can be used to decrease the batch size if the network does not fit.

3.3 Preprocessing

The dataset consisted of JPEG images and the majority of the images were of size 16 megapixels. The images have very different resolutions, colors, aspect ratios, and cropped in various ways. The ConvNets require a fixed input size, so we had to resize/crop all of the images into some fixed dimensions. We experimented with three dimensions: 128×128, 192×192, and 256×256. We followed the same downsampling strategy for all dimensions, which included cropping out the surrounding black areas, sizing down the width, then vertically cropping/letterboxing until the image is square. We have also reformatted the images into a more lossless format which is PNG. Graphicsmagick [17] toolbox was used to perform all the image resizing, cropping, and formatting tasks. An example of the output of the preprocessing stage is illustrated in Fig. 1.

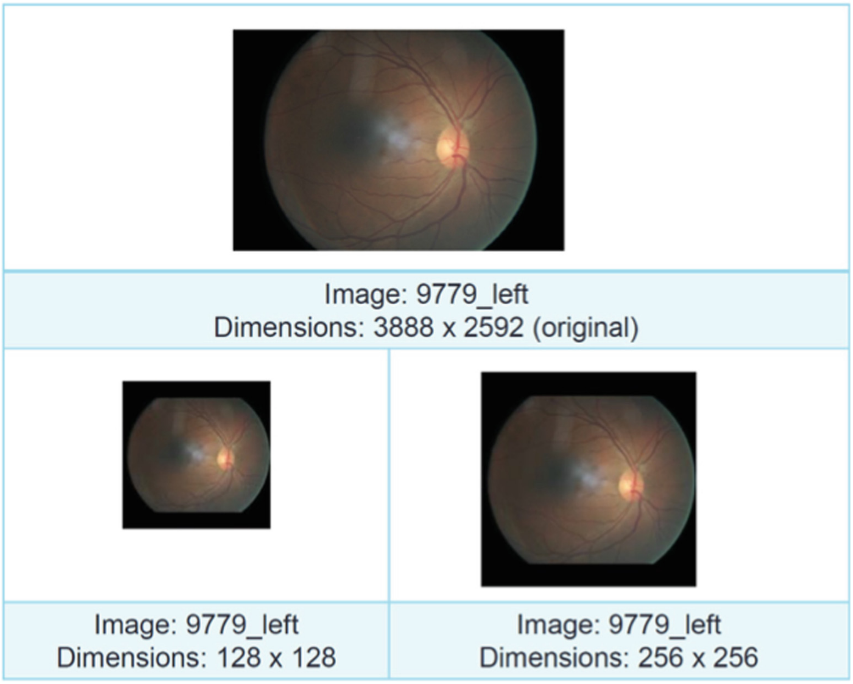


Fig. 1. Example of the preprocessing output

3.4 Data Augmentation

One of the problems of neural networks is that they are extremely powerful. They learn so well that they usually learn something that degrades their performance on other (previously unseen) data. One (made-up) example: the images in the training set are taken by different cameras and have different characteristics. If for some reason, say, the percentage of images of level 3 in dark images is higher than the rest of the images, and the network may start to predict level 3 more often for dark images. One of the solutions to this problem is to enlarge the dataset to minimize the chances of such correlations to happen. This is called data augmentation. Its obvious that if you take an image, zoom it, rotate it, flip it, change the brightness, etc. the level of the disease will not be altered. So it is possible to apply these transformations to the images and obtain much larger and more random training dataset. We applied two main transformations: flipping and color casting. These two transformations helped avoid the problem that some of the images in the dataset were flipped. Flipping: The images were randomly flipped/not flipped along the horizontal axis, and then again for the vertical axis every time it entered the network. This transformation didnt change the image quality or the appearance of the letterbox which was symmetric. Color Casting: this paper [18] was used as a reference to perform the color casting task. It was randomly decided on each channel whether or not to add/subtract a constant.

After that, a centered Gaussian distribution of a standard deviation equal to 10 was used to pick the constant value. This resulted in having 99% of the values which were added being in the range between −30 and 30.

3.5 Network Architecture

Similar to the other top performing solutions in the Kaggle competition, we used deep ConvNets trained on GPUs. ConvNets have been shattering state-of-the-art records on a wide variety of computer vision datasets recently, and they do so without much domain knowledge required The architecture of our network is based on the popular VGGNet [19] from Oxford. Our implementation is adapted from the model implementation of the top 20th entry in the Kaggle DR Competition available on Github [12]. This entry scored a kappa of 0.765 (1st entry scored a kappa of 0.84958) [5]. The architecture consists of 6 convolutional layers where the first layer is the Input layer and three fully connected (FC) layers where the last layer is the Output layer. Each convolutional layer has an LReLu non-linearity, and every convolutional layer (except the first) has dropout with p = 0.1. Each FC layer has dropout with p = 0.5.

4 Results and Discussion

The Kaggle DR competition was scored using Cohens quadratic weighted Kappa function [20]. We used the same metric to evaluate our results and compare it with the other submissions. Cohen's kappa coefficient is a statistic which measures inter-rater agreement for qualitative (categorical) items. It is thought to be a more robust measure than simple percent agreement calculation since takes into account the possibility of the agreement occurring by chance. In our case, Kappa is described as being an agreement between two raters: the agreement between the scores assigned by the human rater (which is unknown to contestants) and the predicted scores. If the agreement is random, the score is close 0 (sometimes it can even be negative). In the case of a perfect agreement, the score is 1. It is quadratic in a sense that, for example, if you predict level 4 for a healthy eye, it is 16 times worse than if you predict level 1. Winners achieved a score more than 0.84. Our best result was around 0.72.

Table 2. Batch size

Batch size	Epochs	Parameters (M)	MB	Kappa
64	200	11.7	31.15	0.697
128	200	11.7	31.15	0.697
256	300	11.7	31.15	0.697

To decide on the appropriate number of epochs which should be used in our experiments, we experimented with a number of epochs from 50 to 400 epochs.

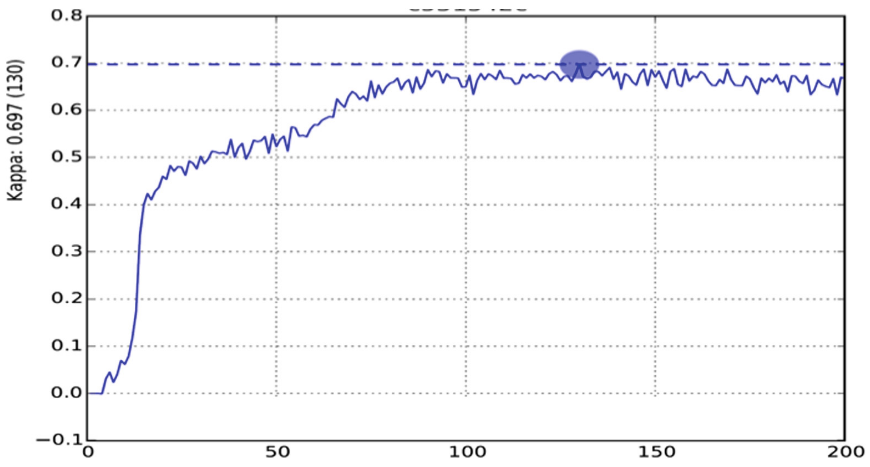


Fig. 2. Kappa when batch size is 64

As we did not see a significant improvement going from 200 to 400 epochs, we stuck with the former for most of our experiments. For all figures in this section, the x-axis represents the number of epochs and the y-axis represents the Kappa score. The label of the y-axis shows the best Kappa score and between brackets the epoch number at which the best Kappa was obtained.

In this experiment, we will study the impact of changing the batch size hyper-parameter on the performance of the model. We will experiment with three batch sizes: 64, 128, and 256. Table 2 and Figs. 2, 3, and 4 describe the network setup used for this experiment and the obtained Kappa results. Our results indicate that increasing the batch size from 64 to 256 doesn significantly affect the Kappa score (δ Kappa = 0.02). However, its clear that the noise and fluctuations were minimized moving from 64 to 256 batch sizes. Besides, it rejects our hypothesis that the gradient decent converges faster with increased batch size. Our results showed that increasing the batch size increased the number of epochs which the gradient needs to converge properly. The reason for this might be that we need to adjust the learning rate as we increase the batch size, but in our case, we have used the same learning rate for all batch sizes. So the practical recommendation is: pick batch sizes that fully leverage the GPU (e.g. until the memory is filled up), and just choose the largest step-size that works well in practice after a few quick trials.

In this experiment, we will study the impact of the number and size of fully connected layers on the model performance. The first test we performed was to decrease the number of fully connected layers from 2 layers to 1 layer (Setup 1 in Table 3). The results of this test are illustrated in Fig. 5. The second test was to decrease the number of units for setup 1 from 2048 to 1024 units (Setup 2 in Table 3). The results of this test are illustrated in Fig. 6. Our results show that the kappa score decreased by only 0.01 when the number of layers was decreased

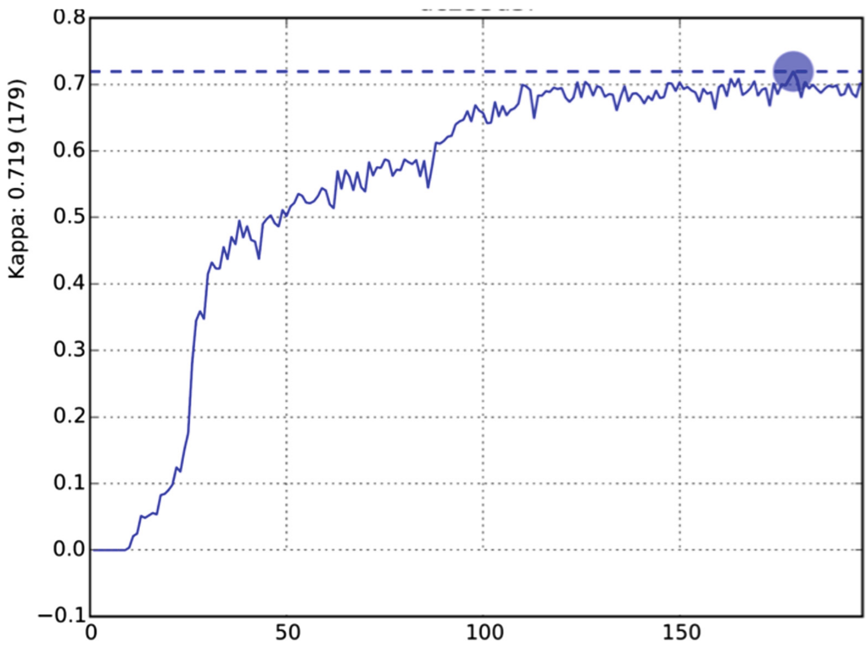


Fig. 3. Kappa when batch size is 128

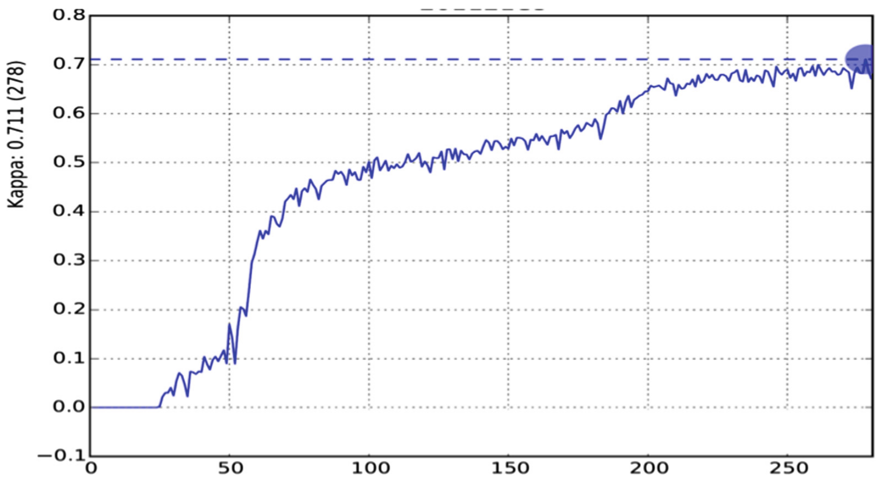


Fig. 4. Kappa when batch size is 256

by half (2 layers to 1 layer), and also by only 0.01 when the number of units was decreased by half (2048 to 1024). These results completely support our second hypotheses that the number and size of fully connected layers do not significantly impact the performance of the model. In addition, the results illustrate that the

Table 3. Number and size of fully connected layers

Title	Reference	Setup1	Setup2
Batch size	128	128	128
Resolution	256 × 256	256 × 256	256 × 256
Epochs	200	200	200
FC layers	2	1	1
Units per FC layer	2048	2048	1024
Parameters (million)	11.17	9.1	4.7
Memory/image (MB)	31.15	31.15	31.15
Execution time (hours)	27	25	15
Best Kappa	0.72	0.71	0.70

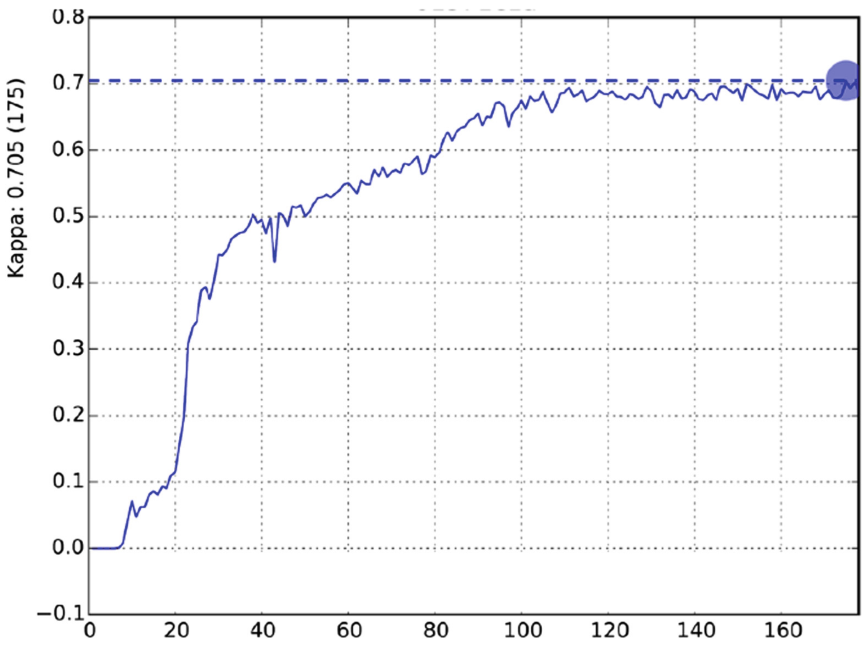


Fig. 5. Kappa score for network setup 1

network complexity was simplified by reducing the number of parameters from 11.17 to 4.7 million, which resulted in reducing the execution time from 27 to 15 h while only losing 2% of the Kappa score.

Increasing the batch size does not necessarily speed up the gradient computations. However, it helps to reduce the fluctuations in the model performance. Further investigation is required to identify how the learning rate and step size should be adjusted with respect to changes in the batch size. We did not notice

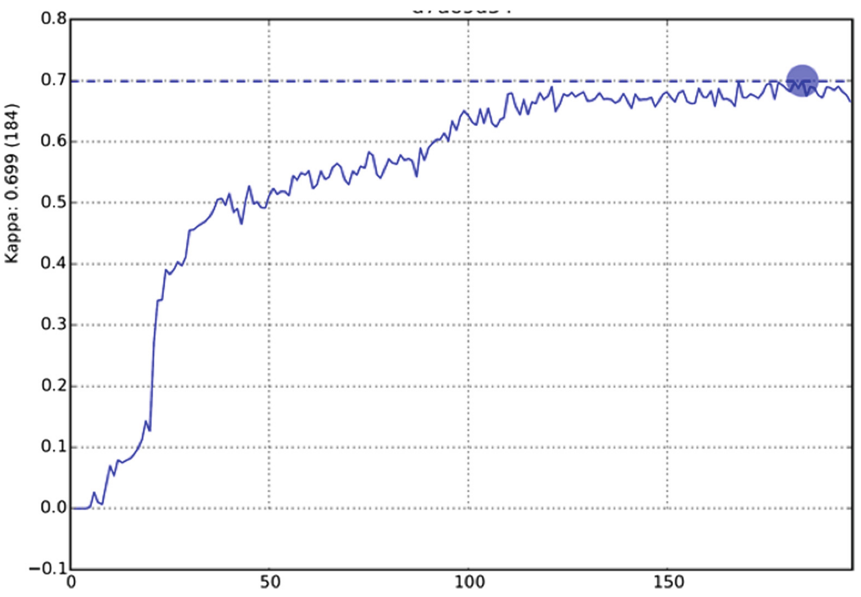


Fig. 6. Kappa score for network setup 2

any significant difference in the Kappa score when increasing the batch size from 64 to 256 (Δ Kappa = 0.02). The number of fully connected layers in the ConvNet architectures does not seem to be very important. The results demonstrated that the difference between having 2 FC layers compared to only 1 FC layer is tiny (Δ Kappa = 0.01). The number of units in the fully connected layers in the ConvNet architectures is also not very important. The results illustrated that the difference between having 1 FC layer with 2048 units compared to 1 FC layer with 1024 units is very little (Δ Kappa = 0.01). Dropout helps fight overfitting. We used 10% probability for all CONV layers and 50% chance for FC layers. The data set for this competition is exclusive in that the images contain important features such as Micro-aneurysms, which are subtle with respect to the size of the retina. We have learned that it is critical for performance to use rather large input images. The maximum image size which we were able to use for training the model was 256 pixels. Given more substantial computing resources, specifically more GPU memory, we would like to try larger image resolutions.

In this work, it is not only important to measure the number of correctly and incorrectly classified tests images, but also to evaluate by how many classes the images were misclassified and to penalize the accuracy score accordingly. Therefore, quadratic Kappa was used as our performance measurement. However, previous work on diabetic retinopathy classification have used different measurements to evaluate the performance of their models such as: accuracy, AUC, specificity, and sensitivity. Therefore, further experimental analysis is required to compare the performance of our model to other state-of-the-art methods. This includes preprocessing our dataset to be ready for experimental analysis with other models, as well as using common performance measurements across all the models in addition to the Kappa (e.g. accuracy, AUC, specificity, and sensitivity).

A number of approaches can be used to improve the performance of our model in the future. For example, multiple models can be merged together within the ConvNet architecture, this is called ensemble learning. Given a single image, we can attempt to classify it several times using the same network, but randomly rotating the images, and/or using independently trained networks. The results of the multiple tests can then be averaged to give a single prediction. This kind of ensemble learning often yields substantial improvements regarding accuracy, although it increases the complexity and computational cost. Even adding somewhat weak networks to an ensemble can improve performance.

Another approach for improving the performance of the model is to create a larger feature vector. We can append more meta-data about the images such as the probability distribution for the person's other eye (left right), the size of the original images, the variance of the original images, and the difference in the preprocessed images.

5 Conclusion

We have shown that ConvNets have the potential to be trained to identify the features of Diabetic Retinopathy in fundus images. The outcomes of the proposed model are auspicious compared to a conventional network topology. In future, we have plans to collect a much cleaner data set from local clinics. The ongoing developments in CNNs allow much deeper networks which could learn better the complex features that this network struggled to learn. Moreover, as data sets are improving continuously, this model can be beneficial to DR clinicians by offering a real-time classifier soon.

Acknowledgment. This work was supported by Sidra Medicine (authors RA and SB), as well as a grant from the Qatar National Research Fund through National Priority Research Program (NPRP) No. 6-249-1-053 (authors SA and MA).

References

1. Boyd, K.: What is Diabetic Retinopathy? https://www.aao.org/eye-health/diseases/what-is-diabetic-retinopathy (2017). Accessed 25 Feb 2017
2. Federation ID: Diabetes: Facts and Figures (2017). http://www.idf.org/about-diabetes/facts-figures. Accessed 25 Feb 2017
3. HMC Expert: Regular Checkups Can Help Prevent Diabetic Retinopathy (2017). https://www.hamad.qa/EN/news/2016/July/Pages/Regular-Checkups-can-help-prevent-Diabetic-Retinopathy-says-HMC-Expert.aspx. Accessed 25 Feb 2017
4. Elshafei, M., Gamra, H., Khandekar, R., Al Hashimi, M., Pai, A., Farouk Ahmed, M.: Prevalence and determinants of diabetic retinopathy among persons 40 years of age with diabetes in Qatar: a community-based survey. Eur. J. Ophthalmol. **21**(1), 39 (2011)

5. Kaggle: Diabetic Retinopathy Detection (2017). https://www.kaggle.com/c/diabetic-retinopathy-detection/data. Accessed 25 Feb 2017
6. Gulshan, V., Peng, L., Coram, M., Stumpe, M.C., Wu, D., Narayanaswamy, A., Venugopalan, S., Widner, K., Madams, T., Cuadros, J., et al.: Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. JAMA **316**(22), 2402 (2016)
7. Haloi, M.: arXiv preprint arXiv:1505.04424 (2015)
8. Wang, S., Yin, Y., Cao, G., Wei, B., Zheng, Y., Yang, G.: Hierarchical retinal blood vessel segmentation based on feature and ensemble learning. Neurocomputing **149**, 708 (2015)
9. Melinščak, M., Prentašić, P., Lončarić, S.: In: VISAPP 2015, 10th International Conference on Computer Vision Theory and Applications (2015)
10. Fraz, M.M., Remagnino, P., Hoppe, A., Uyyanonvara, B., Rudnicka, A.R., Owen, C.G., Barman, S.A.: Blood vessel segmentation methodologies in retinal images - a survey. Comput. Methods Programs Biomed. **108**(1), 407 (2012)
11. Pratt, H., Coenen, F., Broadbent, D.M., Harding, S.P., Zheng, Y.: Convolutional neural networks for diabetic retinopathy. Procedia Comput. Sci. **90**, 200 (2016)
12. Kaggle-dr: My Entry for the Kaggle Diabetic Retinopathy Competition for 20/661 Place (2017). https://github.com/ilyakava/kaggle-dr. Accessed 25 Feb 2017
13. Theano: Theano 0.8.2 Documentation (2017). http://deeplearning.net/software/theano/. Accessed 25 Feb 2017
14. Lasagne: Lightweight Library to Build and Train Neural Networks in Theano (2017). https://github.com/Lasagne/Lasagne. Accessed 25 Feb 2017
15. Pylearn2: Pylearn2 Documentation (2017). http://deeplearning.net/software/pylearn2/. Accessed 25 Feb 2017
16. NAVIDA: CUDA Toolkit Documentation (2017). http://docs.nvidia.com/cuda. Accessed 25 Feb 2017
17. GraphicsMagick: GraphicsMagick Image Processing System (2017). http://www.graphicsmagick.org. Accessed 25 Feb 2017
18. Wu, R., Yan, S., Shan, Y., Dang, Q., Sun, G.: arXiv preprint arXiv:1501.02876 **7**(8) (2015)
19. Simonyan, K., Zisserman, A.: arXiv preprint arXiv:1409.1556 (2014)
20. Cohen, J.: A coefficient of agreement for nominal scales. Educ. Psychol. Meas. **20**(1), 37 (1960)

Compact Representation of Documents Using Terms and Termsets

Dima Badawi[1] and Hakan Altınçay[2(✉)]

[1] Computer Department, Palestine Technical University - Kadoorie, Hebron, Palestine
dbadawi@ptca.edu.ps

[2] Computer Engineering Department, Eastern Mediterranean University, Famagusta, North Cyprus, Mersin 10, Turkey
hakan.altincay@emu.edu.tr

Abstract. In this study, computation of compact document vectors by utilizing both terms and termsets for binary text categorization is addressed. In general, termsets are concatenated with all terms, leading to large document vectors. Selection of a subset of terms and termsets for compact but also effective representation of documents is considered in this study. Two different methods are studied for this purpose. In the first method, combination of terms and termsets in different proportions is evaluated. As an alternative approach, normalized ranking scores of terms and termsets are employed for subset selection. Experiments conducted on two widely used datasets have shown that termsets can effectively complement terms also in cases when small number of features are used to represent documents.

1 Introduction

Document representation is a common task in various text categorization problems such as sentiment classification, emotion recognition and email filtering. Using the words (or, terms) that appear, the main goal is to define discriminative features. Bag-of-words (BOW) is the most frequently used approach where co-occurrence information is not considered. After a set of discriminative terms is computed, a feature is defined for each term as the product of its frequency in the document under concern and its collection frequency factor (CFF) [1]. In general, CFF is computed as a document-independent factor to quantify the discriminative power of the term by analyzing the whole training corpus. Alternatively, co-occurrence of different terms can be utilized to enrich BOW-based representation [2]. The most widely used approach for employing co-occurrence information is *ngrams*[3]. An ngram corresponds to a group of n terms that appear consequently and in ordered form. Bigrams ($n = 2$) and trigrams ($n = 3$) are generally found to improve document representation when considered together with BOW. It should be noted that ngrams of characters are

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 77–84, 2018.
https://doi.org/10.1007/978-3-319-96136-1_7

also studied for some problems such as sentiment classification [4]. As an alternative way of representing co-occurrences, sets of terms (namely, *termsets*) that are not necessarily adjacent is extensively studied [5–7]. Recent experiments conducted on text categorization has proven the potential of termsets for document representation in text classification [8].

In BOW representation, when SVM is used as the classification scheme, it is well known that the best performance scores are achieved when thousands of terms are utilized [8]. More specifically, it is shown that approximately five thousand terms are necessary to achieve the highest performance scores in binary text classification [9]. When ngrams are used together with terms, the dimensionality of feature vectors might be almost doubled. In the case of termsets, millions of additional features can be defined. Recent experiments conducted on text classification have shown that using one thousand 2-termsets may be a good setting for two widely used datasets [8]. As a matter of fact, choosing the most discriminative co-occurrence-based features is highly important [10].

When both terms and ngrams/termsets are used for document depresentation, the tasks of identifying the term-based and co-occurrence-based features are generally done independently from each other. Because of this, some terms may become redundant after adding ngrams/termsets. For instance the bigram "information retrieval" may be more descriptive about the content of the document when compared to "information" or "retrieval. "machine learning" is another typical example [11]. On the other hand, the general practice of concatenating the co-occurrence based features with all terms leads to large document vectors. As a matter of fact, removal of some terms from BOW should also be addressed. When a small set of features need to be defined, selection of the best-fitting set of terms and ngrams/termsets becomes more challenging. In fact, to the best of our knowledge, document representation using both types of features when small number of features are utilized is not extensively investigated. In this study, selection of a small subset of term and termset-based features for compact representation of documents in binary text classification is studied. As the first approach, effectiveness of using terms and termsets in different proportions is addressed, aiming to compute a general rule of combination that can be applied for different text classification problems. Selection of a subset of terms and termsets by taking into account their ranking scores is then considered. Experiments conducted on two widely used datasets namely, 20 Newsgroups and OHSUMED have shown that, using 75% terms and 25% termsets is an acceptable rule of combination for an a priori determined number of features. It is also observed that the best-fitting proportion depends on the number of features to be selected. Moreover, better performance scores than BOW-based representation are achieved by using smaller number of features.

2 Experimental Methodology

The simulations are conducted using the experimental framework that is recently proposed by the authors of this study [8]. In this section, a brief review of its main components is presented.

The documents are firstly grouped as positive and negative classes. For a given category, all documents in that category are labeled as positive whereas the remaining documents are put into the negative class. The documents are then preprocessed to remove stop-words. The Porter's stemming algorithm is applied afterwards. Using χ^2, 5000 top-ranked terms are then selected. The document vectors are cosine normalized so as to eliminate the effect of differences in document lengths on term frequencies. For each of the selected terms, the collection frequency factor is computed using relevance frequency (RF) as [9]

$$CFF = \log_2\left(2 + \frac{A}{max\{1, C\}}\right), \tag{1}$$

where A and C denote the numbers of positive and negative documents which include the term. The weight of each term t_i is calculated as the product of its term frequency (tf_i) and CFF as $tf_i \times CFF_i$.

In this study, termsets of two terms are considered. After constructing all termsets as pairs of all selected terms, they are ranked using a modified form of χ^2 that is defined as [8]

$$\hat{\chi}^2 = \frac{T(\hat{A}\hat{D} - \hat{B}\hat{C})^2}{(\hat{A}+\hat{C})(\hat{B}+\hat{D})(\hat{A}+\hat{B})(\hat{C}+\hat{D})}, \tag{2}$$

where T is the total number of documents in the training corpus. For a given termset, $\hat{A}$ is the number of positive documents which include *either or both* of the members and $\hat{B}$ is the number of remaining positive documents. Similarly, $\hat{C}$ is the number of negative documents which include either or both of the members.

The CFFs of termsets are computed as follows. Consider the termset $\mathbf{t} = \{t_i, t_j\}$. The CFF of $\mathbf{t}$ is defined to be non-zero when at least one of its members appears. In other words, three CFF values are computed for each termset. CFF^{++} is the weight employed when both members appear whereas CFF^{+-} is used when only the first member, t_i appears. Similarly, CFF^{-+} is utilized when only the second term is available. CFF^{++} is defined as

$$CFF^{++} = \log_2\left(2 + \frac{A^{++}}{max\{1, C^{++}\}}\right), \tag{3}$$

where A^{++} and C^{++} respectively denote the numbers of positive and negative documents where both t_i and t_j occur. Similarly,

$$CFF^{+-} = \log_2\left(2 + \frac{A^{+-}}{max\{1, C^{+-}\}}\right) \tag{4}$$

where A^{+-} and C^{+-} respectively denote the numbers of positive and negative documents where t_i occurs but t_j does not occur and,

$$CFF^{-+} = \log_2\left(2 + \frac{A^{-+}}{max\{1, C^{-+}\}}\right) \tag{5}$$

where A^{-+} and C^{-+} respectively denote the numbers of positive and negative documents where t_i does not occur but t_j appears. Taking into account the members which occur in the document under concern, the weight of each termset is computed as the product of the sum of the term frequencies, $(tf_i + tf_j)$ and the corresponding CFF. As a matter of fact, the CFF of a termset depends on the document under concern.

3 Construction of Document Vectors

As the first approach, we studied including terms in different proportions (represented by P_{term}), ranging from $P_{term} = 10\%$ to $P_{term} = 95\%$. When N features are to be selected for document representation, using $P_{term} = 10\%$ means that 10% of N features are the top-ranked terms and 90% of the features are top-ranked termsets. By conducting the experiments on two widely-used datasets, it is aimed to evaluate whether a proportion that can be generalized can be found. The experimental results are also compared with BOW representation, i.e. $P_{term} = 100\%$. For BOW-based document representation, the *CFF* used in this study, i.e. *RF* can be considered as the state-of-the-art [9].

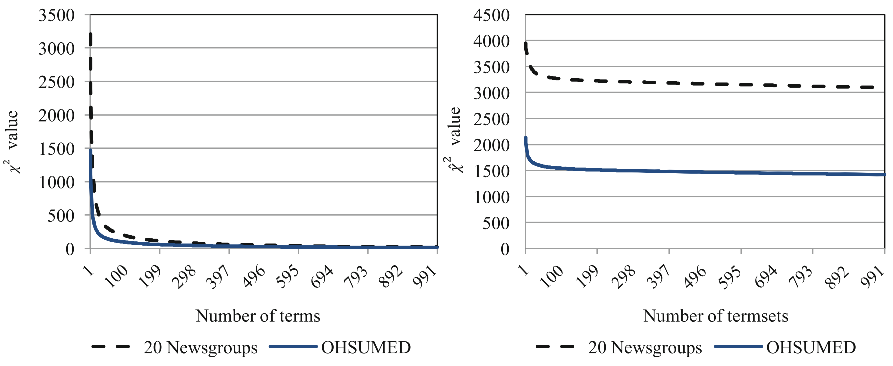


Fig. 1. The χ^2 values of 1000 top-ranked terms (on the left) and $\hat{\chi}^2$ values of 1000 top-ranked termsets (on the right).

As the second selection scheme, the desired number of features are selected by using the ranking scores, i.e. χ^2 and $\hat{\chi}^2$ values. Figure 1 presents the χ^2 values of top-ranked 1000 terms (on the left) and $\hat{\chi}^2$ values of top-ranked 1000 termsets (on the right). It can be seen that $\hat{\chi}^2$ values are much larger. If we select the top-ranked features from a bag of terms and termsets, the feature set will not include any term. Because of this, the ranking scores are firstly normalized using zero-mean and unit-variance normalization.

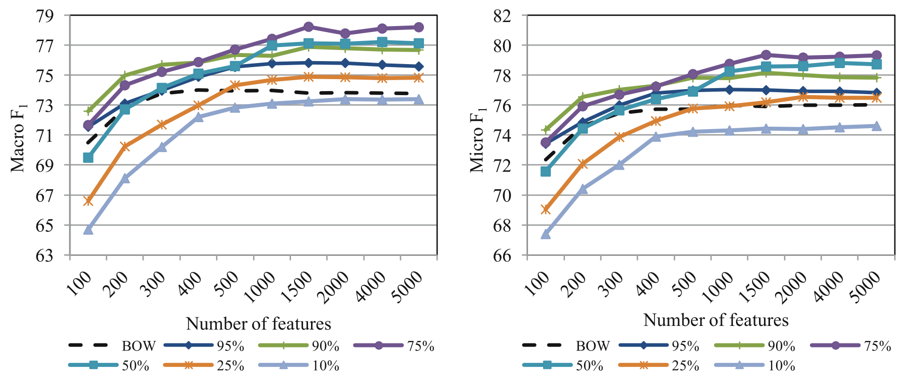


Fig. 2. The macro and micro F_1 scores achieved using top-ranked terms and termsets in different proportions on 20 Newsgroups.

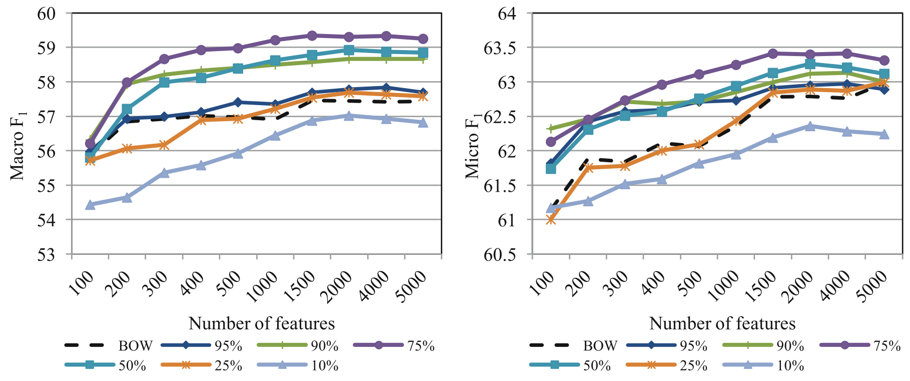


Fig. 3. The macro and micro F_1 scores achieved using top-ranked terms and termsets in different proportions on OHSUMED.

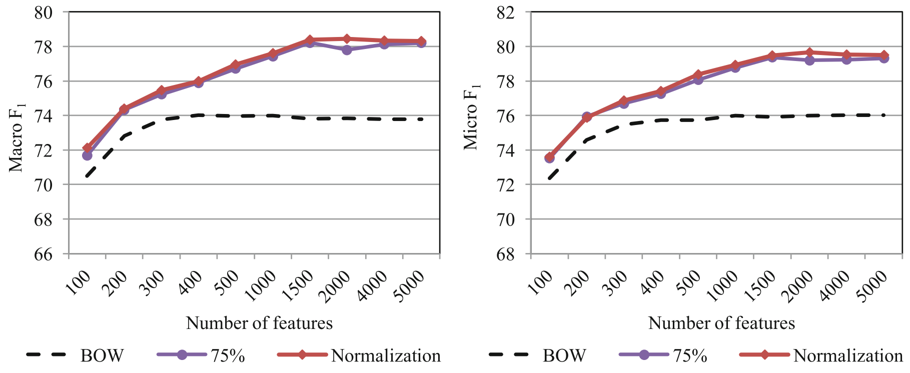


Fig. 4. The macro and micro F_1 scores achieved using top-ranked terms and termsets selected after score normalization on 20 Newsgroups.

4 Experiments

The experiments are conducted on two widely-used datasets, namely 20 Newsgroups and OHSUMED. There are 20 and 23 categories in these datasets, respectively. A classification task is defined for each category by defining the positive class as the set of all documents belonging to the target category and, the macro and micro F_1 scores are reported for each dataset. In all simulations, SVM with linear kernel is used as the classification scheme.

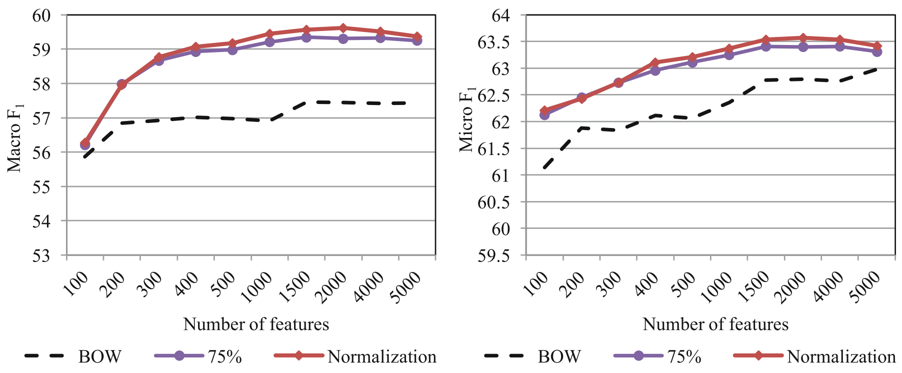


Fig. 5. The macro and micro F_1 scores achieved using top-ranked terms and termsets selected after score normalization on OHSUMED.

Figures 2 and 3 respectively present the F_1 scores achieved for different proportions considered on 20 Newsgroups and OHSUMED. The horizontal axes represent the total number of features used. BOW corresponds to the case when all features are terms, i.e. $P_{term} = 100$. It can be seen that both macro and micro F_1 scores generally improve as P_{term} increases from 10% to 75% and decreases if it is increased further. The best scores are achieved when $P_{term} = 75\%$ and 1500 features are used. This means that 1125 terms and 375 termsets form the best-fitting feature vectors. Even when used in small proportions such as 5%, it can be seen in the figures that termsets enrich the BOW representation. Another important observation is that, when the number of features is small (i.e. smaller than 400 in the case of 20 Newsgroups and smaller than 200 in the case of OHSUMED), the best scores are achieved using $P_{term} = 90\%$. This means that there is a set of discriminative terms that must always be included in the combined feature set. When P_{term} is small, they may be omitted and hence the performance degrades. However, this is a small set when compared to the whole set of 5000 terms.

Figures 4 and 5 present the F_1 scores achieved by ranking terms and termsets after score normalization on 20 Newsgroups and OHSUMED, respectively. The scores achieved by combining in proportions ($P_{term} = 0.75$) is also provided as a reference. It can be seen that ranking after score normalization generally provides

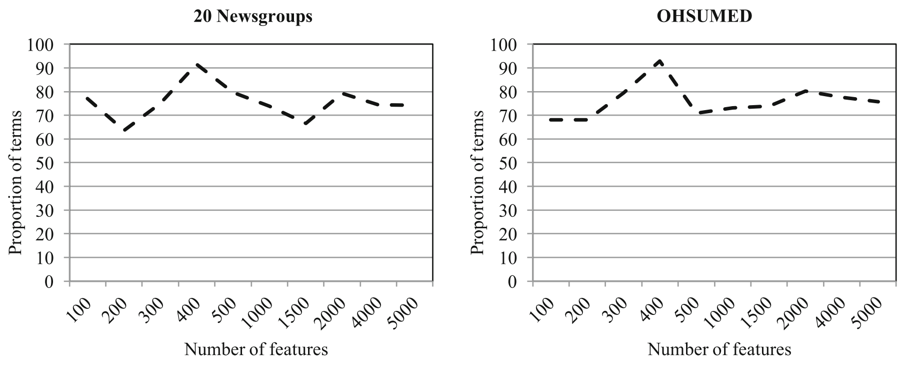


Fig. 6. The average proportion of terms (P_{term}) for each different number of features.

better scores. In order to investigate the P_{term} values corresponding to this approach, we computed the average values over all categories for each different number of features. The results are presented in Fig. 6. Although the best-fitting value depends on the number of features, $70 < P_{term} < 90$ in majority of the cases.

5 Conclusions

It is well known that termsets can contribute to document representation and, when 5000 terms are considered, the performance scores may continue to increase up to the case of 5000 termsets [8]. In this study, it is shown that termsets have a large potential to provide complementary information to terms also in cases when documents need to be represented using smaller number of features. Moreover, significant improvements are achieved when compared to the BOW representation using much smaller number of features.

The experimental results clearly show that the best-fitting document vectors include both terms and termsets. However, $70 < P_{term} < 90$ is in fact an approximate estimate for the best-fitting proportion. As a further research, development of alternative strategies based on the state-of-the-art feature selection techniques which take into account correlations between terms and termsets as well should be addressed.

References

1. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. **24**, 513–523 (1988)
2. Zhang, W., Yoshida, T., Tang, X.: Text classification based on multi-word with support vector machine. Knowl.-Based Syst. **21**(8), 879–886 (2008)
3. Tripathy, A., Agrawal, A., Rath, S.K.: Classification of sentiment reviews using n-gram machine learning approach. Expert Syst. Appl. **57**, 117–126 (2016)

4. Zhai, Z., Xu, H., Kang, B., Jia, P.: Exploiting effective features for chinese sentiment classification. Expert Syst. Appl. **38**, 9139–9146 (2011)
5. Tesar, R., Poesio, M., Strnad, V., Jezek, K.: Extending the single words-based document model: a comparison of bigrams and 2-itemsets. In: Proceedings of the 2006 ACM Symposium on Document Engineering, pp. 138–146. ACM, New York (2006)
6. Zaïane, O.R., Antonie, M.L.: Classifying text documents by associating terms with text categories. In: Proceedings of the 13th Australasian Database Conference ADC 2002, vol. 5, pp. 215–222. Australian Computer Society, Inc. (2002)
7. Figueiredo, F., Rocha, L., Couto, T., Salles, T., Gonçalves, M.A., Meira, W.: Word co-occurrence features for text classification. Inf. Syst. **36**(5), 843–858 (2011)
8. Badawi, D., Altınçay, H.: A novel framework for termset selection and weighting in binary text classification. Eng. Appl. Artif. Intell. **35**, 38–53 (2014)
9. Lan, M., Tan, C.L., Su, J., Lu, Y.: Supervised and traditional term weighting methods for automatic text categorization. IEEE Trans. Pattern Anal. Mach. Intell. **31**(4), 721–735 (2009)
10. Ogura, H., Amano, H., Kondo, M.: Comparison of metrics for feature selection in imbalanced text classification. Expert Syst. Appl. **38**(5), 4978–4989 (2011)
11. Rossi, R.G., Rezende, S.O.: Building a topic hierarchy using the bag-of-related-words representation. In: DocEng, pp. 195–204. ACM, New York (2011)

PATENet: Pairwise Alignment of Time Evolving Networks

Shlomit Gur[1](✉) and Vasant G. Honavar[1,2]

[1] Huck Institutes of the Life Sciences, Pennsylvania State University, University Park, PA 16802, USA
szg180@psu.edu

[2] College of Information Sciences and Technology, Pennsylvania State University, University Park, PA 16802, USA

Abstract. Networks that change over time, e.g. functional brain networks that change their structure due to processes such as development or aging, are naturally modeled by time-evolving networks. In this paper we present PATENet, a novel method for aligning time-evolving networks. PATENet offers a mathematically-sound approach to aligning time evolving networks. PATENet leverages existing similarity measures for networks with fixed topologies to define well-behaved similarity measures for time evolving networks. We empirically explore the behavior of PATENet through synthetic time evolving networks under a variety of conditions.

Keywords: Network science · Multilayer networks · Temporal alignment

1 Introduction

Network science has provided a variety of powerful tools for describing, representing, and analyzing a variety of real-world systems including social networks, the internet, functional brain networks, and biomolecular networks [9, 10]. While many of the tools and techniques of network science, e.g. topological analyses and network alignment, focus on networks with fixed topologies, the structure of networks that represent real-world systems change over time. Such networks are naturally modeled as time-evolving networks (TENs) [11, 14]. TENs can display dynamics on networks (where the network structure does not change over time, but the activity of the nodes does); dynamics of networks (where the activity of nodes does not change but the structure does); and dynamics of and on networks (where both the structure and activity change over time) [3]. The relatively young sub-field of TENs [9] has already yielded a substantial body of work, focusing primarily on models of time-evolving networks and the characterization of network dynamics [14]. However, there is limited work on methods for comparative analyses of TENs.

To motivate the underlying problem, consider experimental subjects who undergo functional magnetic imaging (fMRI) recordings of resting state brain activation at different points in time, e.g. in the context of a longitudinal study of changes in functional connectivity as a function of development, aging, or disease progression [16]. The resulting data from each subject are naturally represented as a temporally ordered sequence of functional connectivity networks. To complicate matters, it may not be straightforward to establish one-to-one correspondence between the recording times across subjects because of differences in the timing of recordings, missed recording sessions, etc. Furthermore, even in the case of subjects with recordings obtained at what appears to be matching time points e.g. age in years, because of differences in the onset and progression of development, aging, or disease, and the trajectories across subjects, the networks at the respective time points may not be comparable. With the exception of [15], which focuses on temporal registration of deforming meshes, to the best of our knowledge, there is no work on aligning (temporally) ordered sequences of networks (OSN). The most closely related body of work focuses on aligning ordered sequences of letters over a finite alphabet e.g. DNA or protein sequences [17], video frames [4], and clinical histories [13]. However, with the exception of methods for aligning sequences of letters [17], the methods used are ad hoc and are not supported by a sound mathematical rationale and hence lack precise mathematical characterization and are not amenable to generalization to other related problem domains.

Against this background, we focus on the problem of aligning a pair of OSNs. Specifically, we describe PATENet, a mathematically sound family of algorithms for aligning a pair of OSNs. PATENet requires as input, in addition to a pair of OSNs to be aligned, a measure of pairwise similarity of fixed topology networks, a monotonically increasing function, and a match threshold. It produces as output an optimal alignment of the given pair of OSNs. Specifically, PATENet generalizes the Smith-Waterman (SW) algorithm [17], a dynamic programming algorithm for aligning two ordered sequences of letters, given a pairwise measure of substitutability of letters and gap penalties. SW produces an optimal local alignment, i.e. aligned segments of the given pair of sequences with the largest cumulative similarity. Conceptually, adapting the SW algorithm to yield a mathematically sound algorithm for aligning a pair of OSNs is straightforward; we replace letters by networks, and replace pairwise substitutability of letters by a well-behaved measure of pairwise similarity of (fixed topology) networks. However, in order for this approach to yield both mathematically sound and practically useful algorithms for aligning OSNs, several challenges need to be addressed; there are a variety of measures of similarity or distance between networks that are tailored [6] to meet the needs of specific applications [7]. We need to adjust such measures so as to ensure that the algorithms that use them for aligning OSNs are mathematically well-behaved. In the current work we also show that the PATENet family of algorithms can be readily extended to align ordered sequences of elements other than networks, provided suitable and well-behaved measures of similarity between elements are available.

2 Preliminaries

We use $G = G(\mathcal{V}, \mathcal{E}_G)$ to denote a network, where $\mathcal{V}$ is its set of nodes and $\mathcal{E}_G$ is its set of edges. We define OSN $\mathcal{G}$ to be a sequence of n networks, $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$, where $\forall 1 \leq i \leq n \in \mathbb{N},\ G_i = (\mathcal{V}, \mathcal{E}_{G_i})$ denotes the i th element of $\mathcal{G}$, which is a snapshot of a TEN at time t_i, and $\forall 1 < i \leq n \in \mathbb{N},\ t_{i-1} < t_i$. We use upper case letters, e.g. H, to denote matrices or networks, lower case letters, e.g. n, to denote scalars, and script letters, e.g. $\mathcal{V}$, to denote sets.

Definition 1. Let $G = (\mathcal{V}, \mathcal{E})$ and $G' = (\mathcal{V}, \mathcal{E}')$ be two networks with the same set of nodes $\mathcal{V}$, and respective sets of edges $\mathcal{E}$ and $\mathcal{E}'$ (either identical or different). A function $s(G, G')$, mapping two graphs to $[0, 1]$, is said to be a well-defined *unsigned* normalized network similarity measure (UNNSM) if it satisfies the following properties (adapted from [12]):

1. Identity property: $s(G, G') \leq s(G, G) = 1 \, \forall \, G, G'$.
2. Symmetry property: $s(G, G') = s(G', G) \, \forall \, G, G'$.
3. Minimum property: $s(G, G') \overset{|\mathcal{V}| \to \infty}{\to} 0$ where WLOG G is the complete network, and G' is the empty network (i.e. $\mathcal{E}^C = \mathcal{E}'$).

Definition 2. Similarly, a function $s'(G, G')$, mapping two graphs to $[-1, 1]$, is said to be a well-defined *signed* normalized network similarity measure (SNNSM) if it satisfied the properties described in definition 1, with the minimum property adjusted to the signed range: $s'(G, G') \overset{|\mathcal{V}| \to \infty}{\to} -1$ (rather than 0).

For simplification purposes we assumed G and G' to have the same set of nodes $\mathcal{V}$. However, if $\mathcal{V}_G \neq \mathcal{V}_{G'}$, where $\mathcal{V}_G$ and $\mathcal{V}_{G'}$ denote the set of nodes of G and G', respectively, then $\mathcal{V} = \mathcal{V}_G \bigcup \mathcal{V}_{G'}$ for the definitions above.

2.1 The Smith-Waterman (SW) Sequence Alignment Algorithm

The SW algorithm is a local sequence alignment algorithm, designed to find pairs of segments with high cumulative degree of similarity between two sequences of amino acids (AAs), $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_m\}$. There are 22 AAs, and the similarity between every pair of AAs is specified by the entries of a 'substitution matrix' $SM \in \mathbb{R}^{22 \times 22}$. The SW algorithm uses dynamic programming to generate a 'scoring matrix' $H = H(A, B) \in \mathbb{R}^{(n+1) \times (m+1)}$, which is defined as follows:

$$\begin{aligned} &\forall 0 \leq i \leq n \in \mathbb{N} \;\; \forall 0 \leq j \leq m \in \mathbb{N}, \; H_{i,0} = H_{0,j} = 0 \\ &\forall 0 < i \leq n \in \mathbb{N} \, \forall 0 < j \leq m \in \mathbb{N}, \\ &H_{i,j} = max\{H_{i-1,j-1} + s(a_i, b_j), max_{1 \leq k \leq i}\{H_{i-k,j} - w_k\}, max_{1 \leq l \leq j}\{H_{i,j-l} - w_l\}, 0\} \end{aligned} \tag{1}$$

Where $s(a_i, b_j)$ is the similarity score between the two AAs $a_i \in A$ and $b_j \in B$, according to SM, and w_k is a value assigned to deletions or insertions of length k. Insertions and deletions refer to cases where an element (or a few) within one sequence is not aligned with an element (or a sequence of elements) within the paired sequence. The length of insertions and deletions is the number of consecutive insertions and/or deletions. $w_1 \in \mathbb{R}$ is referred to as 'gap penalty' and is the value assigned to a gap of length 1, and $w_k = f(w_1, k) \in \mathbb{R}$ is the penalty for a gap of length k, where $f(w_1, k)$ can be affine or linear, for example, in relation to w_1.

Let χ denote the maximum value in H, then $\chi : A \times B \to \mathbb{R}$ is the local alignment score between the two sequences A and B, and is used to reveal the best local alignment, by way of backtracing on H. Starting at a cell holding χ, backtracing is performed on H until a cell holding 0 is reached according to the following logic:

1. If $H_{i,j} = H_{i-1,j-1} + s\left(a_i, b_j\right)$, then a_i is aligned with b_j and the process continues from $H_{i-1,j-1}$.
2. Else if $H_{i,j} = H_{i-1,j} - w$, then a_i has no alignment in B, and the process continues from $H_{i-1,j}$.
3. Else if $H_{i,j} = H_{i,j-l} - w$, then b_j has no alignment in A, and the process continues from $H_{i,j-1}$.

The solution is not guaranteed to be unique; there could be multiple cells in H holding χ, in which case the backtracing process can be initiated at any of these cells, resulting in different, yet equally good, local alignments.

3 PATENet

In this paper we focus on aligning a pair of OSNs. To accommodate OSNs resulting from longitudinal recordings from subjects, we impose the following natural desiderata on the alignments returned by PATENet:

1. Preservation in the alignment of the relative order of elements within the sequences. E.g., if element 3 of the first sequence is aligned with element 5 of the second sequence, element 4 of the first sequence can be aligned only with elements in positions 6 or greater in the second sequence.
2. Accommodation of unaligned elements in both sequences (i.e. aligning two sequences of length n and m, respectively, should not force the alignment of $min(n, m)$ elements).
3. Accommodation of longitudinal gaps (e.g. time points existing in one sequence but missing in the other).

3.1 Alternative Substitution Matrix Construction

The SW algorithm requires a well-defined *SM*, holding both positive values for possible matches and negative values for non-matches. Furthermore, unlike in the case of AA sequences, where the sequence elements are drawn from a fixed alphabet, OSNs can contain arbitrary networks defined over a given set of vertices and edges. Hence, we will adapt existing network similarity measures to define pairwise similarity of elements (networks) in OSNs.

Let $\mathcal{G}$ and $\mathcal{G}'$ be two OSNs with n and m elements, respectively. Let s be a well-defined UNNSM. Finally, let $0 < \varphi < 1 \in \mathbb{R}$ be a threshold on s, where $match\left(G_i, G_j'\right) = \begin{cases} 1, \mathit{if}\ \varphi \leq s\left(G_i, G_j'\right) \\ 0, \mathit{if}\ \varphi > s\left(G_i, G_j'\right) \end{cases}$ $\forall G_i \in \mathcal{G}$, $\forall G_j' \in \mathcal{G}'$, and let $\ell(x)$, ℓ: $[0,1] \to [-1,1]$ be a signed normalized monotonically increasing transform, with $\ell(\varphi) = 0$, $\ell(0) = -1$, and $\ell(1) = 1$. We propose

$$SM_{i,j} = \ell\left(s\left(G_i, G_j^{'}\right)\right) \forall 1 \leq i \leq n \in \mathbb{N} \ \forall 1 \leq j \leq m \in \mathbb{N} \tag{2}$$

to construct an 'alternative substitution matrix' $SM = SM(\mathcal{G}, \mathcal{G}') \in [-1, 1]^{n \times m}$.

For example, for $\alpha = \frac{\varphi}{1-\varphi}$ and $0.5 < \varphi < 1$, $\ell(x) = 1 - \log_\alpha[\alpha^2 + (1 - \alpha^2) \cdot x]$ is such a signed normalized monotonically increasing transform (proof omitted), and along with $DeltaCon\left(G_i, G_j^{'}\right)$ [12] as the well-defined UNNSM (by definition), can be used to construct an alternative SM. Another example includes $\tilde{s}(G, G') = (1 - NSSD(G, G'))$ as the well-defined UNNSM (proof omitted), where $NSSD(G, G')$ is the normalized sum squared difference, and $\tilde{\ell}(\tilde{x}) = \begin{cases} \frac{\tilde{x}-\varphi}{1-\varphi}, if\ \tilde{x} \geq \varphi \\ \frac{\tilde{x}-\varphi}{\varphi}, if\ \tilde{x} < \varphi \end{cases}$ as the signed normalized monotonically increasing transform (proof omitted).

Lemma 1. *Let $\mathcal{G}$ and $\mathcal{G}'$ be two OSNs with n and m elements, respectively, and let s be a well-defined UNNSM, and ℓ: $[0, 1] \to [-1, 1]$ be a signed normalized monotonically increasing transform, as described above. Then $\ell(s(G, G'))$, mapping two graphs to $[-1, 1]$, satisfies the properties of a well-defined SNNSM.*

Proof
Identity: $\ell(\mathrm{s}(\mathrm{G}, \mathrm{G}')) \leq \ell(\mathrm{s}(\mathrm{G}, \mathrm{G})) = \ell(1) = \ell(\max\{\mathrm{s}(\mathrm{G}, \mathrm{G}')\}) = \max\{\ell(\mathrm{s}(\mathrm{G}, \mathrm{G}'))\} = 1$ ■
Symmetry: $\ell\left(s\left(G, G^{'}\right)\right) = \ell\left(s\left(G^{'}, G\right)\right)$ ■
Minimum: $\ell\left(s\left(G, G^{'}\right)\right) \overset{|\mathcal{V}| \to \infty}{\to} \ell(0) = \ell\left(min\left\{s\left(G, G^{'}\right)\right\}\right) = min\left\{\ell\left(s\left(G, G^{'}\right)\right)\right\} = -1$ ■

3.2 From SW to PATENet

The SW algorithm meets the first two desiderata of PATENet (preservation of temporal order and accommodation of possible unaligned elements in both sequences). To satisfy the third desideratum (accommodation of longitudinal gaps), we set the gap penalty to zero. Therefore, for an alternative SM, following the construction described above, the scoring matrix of PATENet $\tilde{H} = \tilde{H}\left(\mathcal{G}, \mathcal{G}^{'}\right) \in \mathbb{R}^{(n+1)\times(m+1)}$, hereafter referred to as 'OSN scoring matrix', is specified as follows:

$$\begin{aligned} &\forall 0 \leq i \leq n \in \mathbb{N} \ \forall 0 \leq j \leq m \in \mathbb{N}, \ \tilde{H}_{i,0} = \tilde{H}_{0,j} = 0 \\ &\forall 0 < i \leq n \in \mathbb{N} \ \forall 0 < j \leq m \in \mathbb{N}, \\ &\tilde{H}_{i,j} = max\left\{\tilde{H}_{i-1,j-1} + SM_{i,j}, max_{1 \leq k \leq i}\left\{\tilde{H}_{i-k,j}\right\}, max_{1 \leq l \leq j}\left\{\tilde{H}_{i,j-l}\right\}, 0\right\} \end{aligned} \tag{3}$$

Lemma 2. *Let $\mathcal{G}$ and $\mathcal{G}^{'}$ be two OSNs with n and m elements, respectively. Let $\tilde{H} \in \mathbb{R}^{(n+1)\times(m+1)}$ be their OSN scoring matrix, then:*

(2.1) $\forall 1 \le i \le n \in \mathbb{N}$ *and* $\forall 1 \le j \le m \in \mathbb{N}$, $\tilde{H}_{i-1,j} \le \tilde{H}_{i,j}$
(2.2) $\forall 1 \le i \le n \in \mathbb{N}$ *and* $\forall 1 \le j \le m \in \mathbb{N}$, $\tilde{H}_{i,j-1} \le \tilde{H}_{i,j}$

Proof

(2.1) $\tilde{H}_{i,j} = max\{\tilde{H}_{i-1,j-1} + SM_{i,j}, max_{1 \le k \le i}\{\tilde{H}_{i-k,j}\}, max_{1 \le l \le j}\{\tilde{H}_{i,j-l}\}, 0\} \ge max_{1 \le k \le i}\{\tilde{H}_{i-k,j}\} \ge \tilde{H}_{i-1,j}$ ∎
(2.2) $\tilde{H}_{i,j} = max\{\tilde{H}_{i-1,j-1} + SM_{i,j}, max_{1 \le k \le i}\{\tilde{H}_{i-k,j}\}, max_{1 \le l \le j}\{\tilde{H}_{i,j-l}\}, 0\} \ge max_{1 \le l \le j}\{\tilde{H}_{i,j-l}\} \ge \tilde{H}_{i,j-1}$ ∎

Lemma 3. *Let $\mathcal{G}$ and $\mathcal{G}'$ be two OSNs with n and m elements, respectively. Let $\tilde{H} \in \mathbb{R}^{(n+1)\times(m+1)}$ be their OSN scoring matrix, then:*

(3.1) $\forall 1 \le i \le n \in \mathbb{N}$ *and* $\forall 1 \le j \le m \in \mathbb{N}$, $\underset{1 \le k \le i}{max}\{\tilde{H}_{i-k,j}\} = \tilde{H}_{i-1,j}$
(3.2) $\forall 1 \le i \le n \in \mathbb{N}$ *and* $\forall 1 \le j \le m \in \mathbb{N}$, $\underset{1 \le l \le j}{max}\{\tilde{H}_{i,j-l}\} = \tilde{H}_{i,j-1}$

Proof
Intuitive based on Lemma 2 ∎

Therefore, the OSN scoring matrix $\tilde{H}$ of PATENet is equivalent to:

$$\begin{array}{l} \forall 0 \le i \le n \in \mathbb{N}\ \forall 0 \le j \le m \in \mathbb{N},\ \tilde{H}_{i,0} = \tilde{H}_{0,j} = 0 \\ \forall 0 < i \le n \in \mathbb{N}\ \forall 0 < j \le m \in \mathbb{N}, \\ \tilde{H}_{i,j} = max\{\tilde{H}_{i-1,j-1} + SM_{i,j}, \tilde{H}_{i-1,j}, \tilde{H}_{i,j-1}, 0\} \end{array} \tag{4}$$

Lemma 4. *Let $\mathcal{G}$ and $\mathcal{G}'$ be two OSNs with n and m elements, respectively. Let $\tilde{H} = \tilde{H}\left(\mathcal{G}, \mathcal{G}'\right) \in \mathbb{R}^{(n+1)\times(m+1)}$ and $\tilde{H}' = \tilde{H}\left(\mathcal{G}', \mathcal{G}\right) \in \mathbb{R}^{(m+1)\times(n+1)}$ be their OSN scoring matrices, and let SM and SM′ be the alternative substitution matrices of $\tilde{H}$ and $\tilde{H}'$, repectively. Then $\forall 1 \le i \le n \in \mathbb{N}$ and $\forall 1 \le j \le m \in \mathbb{N}$: (4.1) $SM_{i,j} = SM'_{j,i}$ and (4.2) $\tilde{H}_{i,j} = \tilde{H}'_{j,i}$.*

Proof

(4.1) $\forall 1 \le \mathrm{i} \le \mathrm{n} \in \mathbb{N},\ \forall 1 \le \mathrm{j} \le \mathrm{m} \in \mathbb{N}:\ \mathrm{SM_{i,j}} = \ell\left(\mathrm{s}\left(\mathrm{G_i}, \mathrm{G'_j}\right)\right) = \ell\left(\mathrm{s}\left(\mathrm{G'_j}, \mathrm{G_i}\right)\right) = \mathrm{SM'_{j,i}}$ ∎
(4.2) For $j = i = 1$: $\tilde{H}_{1,1} = max\{\tilde{H}_{0,0} + SM_{1,1}, \tilde{H}_{0,1}, \tilde{H}_{1,0}, 0\} = max\{SM_{1,1}, 0\} = max\left\{SM'_{1,1}, 0\right\} = \tilde{H}'_{1,1}$.

For $j = 1$, $\forall 2 \le i \le n \in \mathbb{N}$, we can safely assume $\tilde{H}_{i-1,1} = \tilde{H}'_{1,i-1}$ for induc tion: $\tilde{H}_{i,1} = max\{\tilde{H}_{i-1,0} + SM_{i,1}, \tilde{H}_{i-1,1}, \tilde{H}_{i,0}, 0\} = max\{SM_{i,1}, \tilde{H}_{i-1,1}, 0\} = max\left\{SM'_{1,i}, \tilde{H}'_{1,i-1}, 0\right\} = \tilde{H}'_{1,i}$.

For $2 \le j = k \in \mathbb{N}$ and $i = 1$: $\tilde{H}_{1,k} = max\{\tilde{H}_{0,k-1} + SM_{1,k}, \tilde{H}_{0,k}, \tilde{H}_{1,k-1}, 0\} = max\{SM_{1,k}, \tilde{H}_{1,k-1}, 0\} = max\{SM_{1,k}, max\{SM_{1,k-1}, \tilde{H}_{1,k-2}, 0\}, 0\} = max\{SM_{1,k}, SM_{1,k-1}, \tilde{H}_{1,\,k-2}, 0\} = \cdots = max\{SM_{1,k}, SM_{1,k-1}, \ldots, SM_{1,2}, SM_{1,1}, 0\} \ldots = max\left\{SM'_{k,1}, SM'_{k-1,1}, \ldots, SM'_{2,1}, SM'_{1,1}, 0\right\} = \tilde{H}'_{k,1}$.

For $2 \le j = k \in \mathbb{N}$, $\forall 1 \le i \le n \in \mathbb{N}$, we can safely assume $\tilde{H}_{i,k-1} = \tilde{H}'_{k-1,i}$ as well as $\tilde{H}_{i-1,k} = \tilde{H}'_{k,i-1}$ and therefore also $\tilde{H}_{i-1,k-1} = \tilde{H}'_{k-1,i-1}$ for induction: $\tilde{H}_{i,k} = max\{\tilde{H}_{i-1,k-1} + SM_{i,k}, \tilde{H}_{i-1,k}, \tilde{H}_{i,k-1}, 0\} = max\left\{\tilde{H}'_{k-1,i-1} + SM'_{k,i}, \tilde{H}'_{k,i-1}, \tilde{H}'_{k-1,i}, 0\right\} = \tilde{H}'_{k,i}$ ∎

Lemma 5. *Let $\mathcal{G}$ and $\mathcal{G}'$ be two OSNs with n and m elements, respectively. Let $\tilde{H} = \tilde{H}\left(\mathcal{G}, \mathcal{G}'\right) \in \mathbb{R}^{(n+1)\times(m+1)}$ be their OSN scoring matrix, and let SM be its alternative substitution matrix. Then the alignment score $\tilde{\chi} = max\{\tilde{H}\}$[1] is equivalent to $\sum_{i=1}^{n} \sum_{j=1}^{m} \left[\rho\left(G_i, G'_j\right) \cdot SM_{i,j}\right]$, where $\rho\left(G_i, G'_j\right) = \begin{cases} 1, \text{ if } \left(G_i, G'_j\right) \text{are aligned with each other} \\ 0, \text{otherwise} \end{cases}$.*

Proof
By definition of the SW algorithm, $\forall 1 \le i \le n \in \mathbb{N}, \forall 1 \le j \le m \in \mathbb{N}, \tilde{H}_{i,j} =$ the maximum similarity of two segments ending in G_i and G'_j. The similarity score of the alignment is the sum of similarity scores between every pair of aligned elements and weights of all insertions and deletions in the alignment. Since $w_1 = 0$ in PATENet, the weights of all insertions and deletions is always 0, leaving only the sum of similarity scores between every pair of aligned elements, which can be written as: $\tilde{\chi} = \sum_{i=1}^{n} \sum_{j=1}^{m} \left[\rho\left(G_i, G'_j\right) \cdot SM_{i,j}\right]$, where $\rho\left(G_i, G'_j\right) = \begin{cases} 1, \text{ if } \left(G_i, G'_j\right) \text{are aligned with each other} \\ 0, \text{otherwise} \end{cases}$ ∎

3.3 OSN Alignment Score

Alignment of elements across a pair of OSNs may be informative by itself and reveal temporally-preserved similarities between the two OSNs. However, another concept worth borrowing from sequence alignment is that of the alignment score $\tilde{\chi} = max\{\tilde{H}\}$.

Lemma 6. *An OSN alignment score $\tilde{\chi} = max\{\tilde{H}\}$ satisfies properties that are similar to those of a well-defined UNNSM, except for the normalization-related upper bound. Identity property[2]: $\tilde{\chi}\left(\mathcal{G}, \mathcal{G}'\right) \le \tilde{\chi}(\mathcal{G}, \mathcal{G})\ \forall \mathcal{G}, \mathcal{G}'$; Symmetry property: $\tilde{\chi}\left(\mathcal{G}, \mathcal{G}'\right) = \tilde{\chi}\left(\mathcal{G}', \mathcal{G}\right)\ \forall \mathcal{G}, \mathcal{G}'$; Minimum property: $\tilde{\chi}\left(\mathcal{G}, \mathcal{G}'\right) \overset{|\mathcal{V}|\to\infty}{\to} 0$ where WLOG $\mathcal{G}$ is the complete OSN, and $\mathcal{G}'$ is the empty OSN (i.e. $\forall 1 \le i \le n \in \mathbb{N}, \forall 1 \le j \le m \in \mathbb{N}, \mathcal{E}^C_{G_i} = \mathcal{E}_{G'_j}$).*

[1] Notice that $\tilde{\chi} : \mathcal{G} \times \mathcal{G}' \to \mathbb{R}$.

[2] Notice that $\tilde{\chi}(G, G) = 1$ is not required, as the alignment score has no upper bound.

Proof
Based on Lemma 5, $\tilde{\chi}(\mathcal{G},\mathcal{G}') = max\{\tilde{H}\} = \sum_{i=1}^{n}\sum_{j=1}^{m}\left[\rho\left(G_i, G_j'\right)\cdot SM_{i,j}\right]$, where $\rho\left(G_i, G_j'\right) = \begin{cases} 1, \mathit{if}\left(G_i, G_j'\right) \mathit{are\ aligned\ with\ each\ other} \\ 0, \mathit{otherwise}\end{cases}$. Identity: $\tilde{\chi}(\mathcal{G},\mathcal{G}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left[\rho\left(G_i, G_j\right)\cdot SM_{i,j}\right] = \sum_{i=1}^{n}[1\cdot 1] = n$ and $\tilde{\chi}\left(\mathcal{G},\mathcal{G}'\right) = \sum_{i=1}^{n}\sum_{j=1}^{m}\left[\rho\left(G_i, G_j'\right)\cdot SM_{i,j}\right] \leq \sum_{i=1}^{n}[1\cdot 1] = n = \tilde{\chi}(\mathcal{G},\mathcal{G})$ ∎

Based on Lemma 4, if $\tilde{H} = \tilde{H}\left(\mathcal{G},\mathcal{G}'\right) \in \mathbb{R}^{(n+1)\times(m+1)}$ and $\tilde{H}' = \tilde{H}\left(\mathcal{G}',\mathcal{G}\right) \in \mathbb{R}^{(m+1)\times(n+1)}$, then $\forall 1 \leq i \leq n \in \mathbb{N}, \forall 1 \leq j \leq m \in \mathbb{N},\ \tilde{H}_{i,j} = \tilde{H}'_{j,i}$. Symmetry: $\tilde{\chi}\left(\mathcal{G},\mathcal{G}'\right) = max\{\tilde{H}\} = max_{1\leq i\leq n, 1\leq j\leq m}\{\tilde{H}_{i,j}\} = max_{1\leq j\leq m, 1\leq i\leq n}\left\{\tilde{H}'_{j,i}\right\} = max\{\tilde{H}'\} = \tilde{\chi}\left(\mathcal{G}',\mathcal{G}\right)$ ∎

Minimum: $\forall 1 \leq i \leq n \in \mathbb{N},\ \forall 1 \leq j \leq m \in \mathbb{N},\quad s\left(G_i, G_j'\right) \overset{|\mathcal{V}|\to\infty}{\to} 0 \Rightarrow \rho\left(G_i, G_j'\right) \overset{|\mathcal{V}|\to\infty}{\to} 0 \Rightarrow \tilde{\chi}(\mathcal{G},\mathcal{G}') = \sum_{i=1}^{n}\sum_{j=1}^{m}\left[\rho\left(G_i, G_j'\right)\cdot SM_{i,j}\right] \overset{|\mathcal{V}|\to\infty}{\to} \sum_{i=1}^{n}\sum_{j=1}^{m}[0] = 0$ ∎

We observe that PATENet can be used to extend the UNNSM used for constructing SM, into an unsigned normalized order-aware OSN similarity measure. Let $\mathcal{G}$ and $\mathcal{G}'$ be two OSNs with n and m elements, respectively. Let $\tilde{H} = \tilde{H}\left(\mathcal{G},\mathcal{G}'\right) \in \mathbb{R}^{(n+1)\times(m+1)}$ be the corresponding OSN scoring matrix, and let SM be its alternative substitution matrix. Let s be the well-defined UNNSM used for constructing SM, and $\rho\left(G_i, G_j'\right) = \begin{cases} 1, \mathit{if}\left(G_i, G_j'\right) \mathit{are\ aligned\ with\ each\ other} \\ 0, \mathit{otherwise}\end{cases}$.

Then $\mathcal{k}: \mathcal{G}\times\mathcal{G}' \to [0,1]$ can be defined as

$$\mathcal{k}(\mathcal{G},\mathcal{G}') = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}\left[\rho\left(G_i, G'_j\right)\cdot s\left(G_i, G'_j\right)\right]}{max\left\{\sum_{i=1}^{n}\sum_{j=1}^{m}\rho\left(G_i, G'_j\right), 1\right\}} \tag{5}$$

which is hereafter referred to as an 'OSN similarity score'.[3]

Lemma 7. *An OSN similarity score $\mathcal{k}(\mathcal{G},\mathcal{G}')$ satisfies identity, symmetry and minimum properties, similar to those that hold for UNNSM. Identity property: $\mathcal{k}(\mathcal{G},\mathcal{G}') \leq \mathcal{k}(\mathcal{G},\mathcal{G}) = 1\ \forall\mathcal{G},\mathcal{G}'$; Symmetry property: $\mathcal{k}(\mathcal{G},\mathcal{G}') = \mathcal{k}(\mathcal{G}',\mathcal{G})\ \forall\mathcal{G},\mathcal{G}'$; Minimum property: $\mathcal{k}(\mathcal{G},\mathcal{G}') \xrightarrow{|\mathcal{V}|\to\infty} 0$ where WLOG $\mathcal{G}$ is the complete OSN, and $\mathcal{G}'$ is the empty OSN (i.e. $\forall 1 \leq i \leq n \in \mathbb{N}, \forall 1 \leq j \leq m \in \mathbb{N}, \mathcal{E}_{G_i}^{C} = \mathcal{E}_{G_j'}$).*

[3] Notice that the OSN similarity score measures similarity in the context of the locally aligned segments of the sequences. That is, if OSNs $\mathcal{G}$ and $\mathcal{G}'$ have k elements aligned with average element-wise similarity of h, whether $k = min(n,m)$ or $k < min(n,m)$, $\mathcal{k}(\mathcal{G},\mathcal{G}') = h$. Additionally, if OSNs $\mathcal{G}$ and $\mathcal{G}'$ have one element aligned with element-wise similarity of 1.0, while OSNs $\mathcal{G}$ and $\mathcal{G}''$ have four elements aligned with each element-wise similarity being 0.9, $\mathcal{k}(\mathcal{G},\mathcal{G}') > \mathcal{k}(\mathcal{G},\mathcal{G}'')$ (but $\tilde{\chi}(\mathcal{G},\mathcal{G}') < \tilde{\chi}(\mathcal{G},\mathcal{G}'')$).

Proof
WLOG, we assume $n \leq m$. Identity:

$$\mathscr{k}(\mathcal{G},\mathcal{G}) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\left[\rho(G_i,G_j)\cdot s(G_i,G_j)\right]}{max\left\{\sum_{i=1}^{n}\sum_{j=1}^{n}\rho(G_i,G_j),1\right\}} = \frac{\sum_{i=1}^{n}\left[1\cdot s(G_i,G_i)\right]}{max\left\{\sum_{i=1}^{n}1,1\right\}} = \frac{n}{n} = 1 \text{ and}$$

$$\mathscr{k}(\mathcal{G},\mathcal{G}') = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}\left[\rho(G_i,G'_j)\cdot s(G_i,G'_j)\right]}{max\left\{\sum_{i=1}^{n}\sum_{j=1}^{m}\rho(G_i,G'_j),1\right\}} \leq \frac{\sum_{i=1}^{n}[1\cdot 1]}{max\left\{\sum_{i=1}^{n}1,1\right\}} \leq \frac{n}{max\{n,1\}} = \frac{n}{n} = 1 = \mathscr{k}(\mathcal{G},\mathcal{G})$$

∎

Symmetry:

$$\mathscr{k}(\mathcal{G},\mathcal{G}') = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}\left[\rho(G_i,G'_j)\cdot s(G_i,G'_j)\right]}{max\left\{\sum_{i=1}^{n}\sum_{j=1}^{m}\rho(G_i,G'_j),1\right\}} = \frac{\sum_{j=1}^{m}\sum_{i=1}^{n}\left[\rho(G'_j,G_i)\cdot s(G'_j,G_i)\right]}{max\left\{\sum_{j=1}^{m}\sum_{i=1}^{n}\rho(G'_j,G_i),1\right\}} = \mathscr{k}(\mathcal{G}',\mathcal{G})$$

∎

Minimum: $\forall 1 \leq i \leq n \in \mathbb{N},\ \forall 1 \leq j \leq m \in \mathbb{N},$

$$s(G_i,G'_j) \xrightarrow{|\mathcal{V}|\to\infty} 0 \implies \rho(G_i,G'_j) \xrightarrow{|\mathcal{V}|\to\infty} 0 \implies \mathscr{k}(\mathcal{G},\mathcal{G}') = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}\left[\rho(G_i,G'_j)\cdot s(G_i,G'_j)\right]}{max\left\{\sum_{i=1}^{n}\sum_{j=1}^{m}\rho(G_i,G'_j),1\right\}} \xrightarrow{|\mathcal{V}|\to\infty} \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}[0]}{max\left\{\sum_{i=1}^{n}\sum_{j=1}^{m}0,1\right\}} = \frac{0}{1} = 0$$

∎

4 Experiments

We now proceed to describe a set of experiments that explore the behavior of PATENet under a variety of conditions. Because "ground truth" alignments for real-world OSNs are unavailable, we generated synthetic OSNs for this purpose. Although PATENet has three user-defined parameters, we experimented with different match thresholds, while keeping the other two parameters (a well-defined UNNSM and a signed normalized monotonically increasing transform) constant, as they are more application- and domain-specific.

4.1 Empirical Design

We experimented with PATENet with a substitution matrix based on DeltaCon [12] and a logarithmic signed normalized monotonically increasing transform function ($\ell(x) = 1 - \log_{\alpha}[\alpha^2 + (1-\alpha^2)\cdot x]$ where $\alpha = \frac{\varphi}{1-\varphi}$ for $0.5 < \varphi < 1$). DeltaCon assesses node affinities similarity between two undirected networks with known node correspondence. It is a well-defined UNNSM (by definition).

To examine the robustness of PATENet to noise in the data, we corrupted one of the OSNs - containing otherwise identical subset of (in our experiments with synthetic data, six) elements in the OSNs to be aligned - with different levels of Gaussian noise added to the edge weights. Since PATENet uses a static match threshold, we also examined the interaction between the effect of noise on PATENet's performance and the choice of match threshold φ. We experimented with $\varphi = \{0.51, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90\}$ and Gaussian noise with $\mu = 0$ and $\sigma = \{0.1, 0.3, 0.5, \cdots, 3.9\}$.

Performance of alignment was evaluated using ‘goodness of alignment’, defined as the percentage of elements with known ground-truth match (based on the construction of the OSNs) that were aligned with their ground-truth matches.

4.2 Synthetic Data Generation

We constructed three sets of synthetic data: (1) random dynamic OSNs, (2) Barabasi-Albert (BA) [1] dynamic OSNs, and (3) Dorogovtsev-Mendes (DM) [5] dynamic OSNs. The BA and DM models describe evolving, rather than dynamic, networks, hence we adapted only the edge addition/removal portions of these models. For each dataset we examined three temporal conditions: linear, a single change in trend, and two changes in trend. The resulting OSNs consisted of 25 elements each, starting from an undirected random network with 50 nodes and a connectivity rate of $\sim$0.12 (141 edges out of possible 1225). We use O to denote such a 25-element OSN. We further experimented with the percent of edges added/removed from one element to the next in O, using one of four percentages: 1%, 2%, 4% or 8%.

Random dynamic OSNs were generated as follows: element 1 was generated using the Erdos-Renyi (ER) model [8]. In case of linear Os, edges were added at random to generate elements 2–25 (see Fig. 1A). Single trend change Os were generated by adding edges at random to generate elements 2–13, and then removing edges at random to generate elements 14–25 (see Fig. 1B). For Os with two trend changes, elements 2–9 were generated by adding edges at random, elements 10–17 were generated by removing edges at random, and elements 18–25 were generated by adding edges at random (see Fig. 1C).

BA and DM dynamic OSNs were generated in a manner similar to random dynamic OSNs, with the following changes: the BA model [2] with 50 nodes and $n = 3$ (resulting in 141 edges out of a possible 1225, similarly to the ER-based element 1) was used to generate element 1, and edges to be added were selected based on the corresponding model. Edge removal is done at random according to both models.

In any OSN O, 12 (roughly half) of the elements were selected at random and kept in order to make up a new OSN, denoted by M. Half (six) of the elements selected for M were then removed from O to generate a new OSN with 19 elements, denoted by O'. Consequently, any pair of OSNs (O', M) constructed according to the preceding procedure, shares six random elements, and $M \nsubseteq O'$ and $O' \nsubseteq M$ (see Fig. 1D). Gaussian noise (see Sect. 4.1) was added only to M prior to alignment.

4.3 Results

In all three synthetic datasets experiments revealed a similar relationship between the performance of PATENet, user-specified threshold φ and the added Gaussian noise (see Fig. 2). For lower values of φ, PATENet showed a high degree of noise tolerance, significantly outperforming random alignment over a broad range of Gaussian noise levels. As φ increased, so did PATENet’s susceptibility to noise, but for tolerable levels of noise, its performance was similar or better, as compared to PATENet with lower φ for the same noise level. We conclude that the choice of φ affects multiple aspects of the performance of PATENet in the presence of noise.

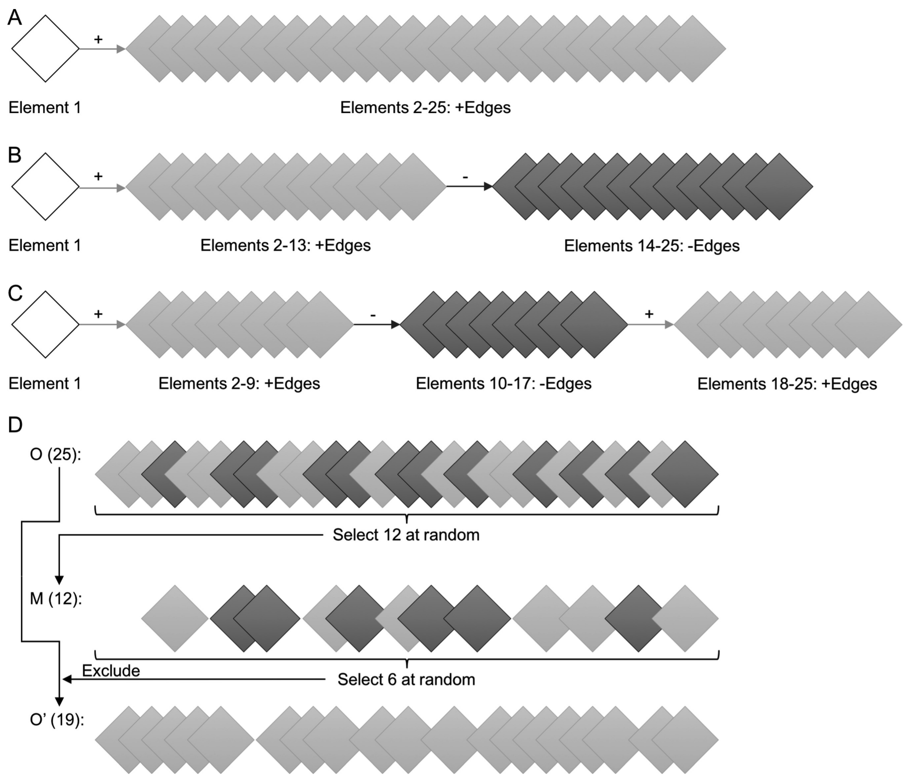


Fig. 1. Synthetic data generation. (A–C) Generation of a linear OSN (A), a single trend change OSN (B) and an OSN with 2 trend changes (C). All OSNs Os (A–C) consist of 25 elements, the first element (white) being a random graph (ER for random dynamics OSNs and BA for BA and DM OSNs). Rectangles represent elements, with light gray indicating increase trend (edges added between elements) and dark gray indicating decrease trend (edges removed between elements); + edges are added between elements; - edges are removed between elements. (D) Generation of OSNs M and O' from OSN O. In dark gray are the elements selected according to the corresponding description in the text. Starting from 25 elements in O, 12 elements are selected at random to create M, six of which are removed from the copy of O to O' (resulting in 19 elements in O').

5 Discussion

5.1 Additional Considerations and Future Directions

In real world OSN data, e.g. those derived from longitudinal studies of functional brain connectivity networks, at present, there are no effective approaches to estimating the noise level in the data. Our results demonstrate a tradeoff between PATENet's resistance to noise and performance with low levels of noise as a function of the choice of match threshold. Hence, in practical settings, it might be worth exploring a probabilistic combination of different match thresholds.

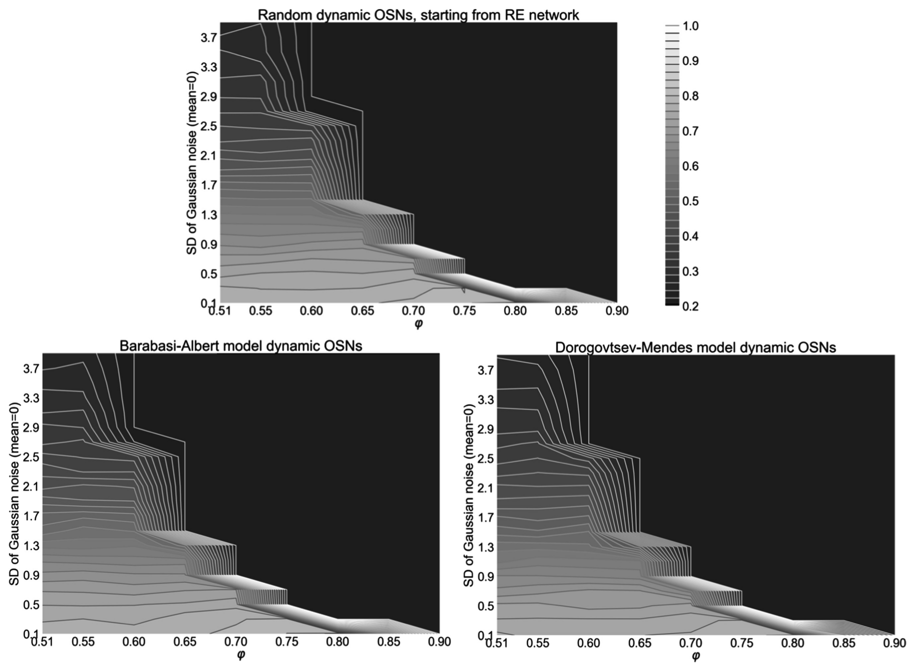


Fig. 2. Effect of noise and match threshold on PATENet's performance. Goodness of alignment of PATENet with the synthetic data as a function of added Gaussian noise (vertical axis) and match threshold φ (horizontal axis). Goodness of alignment was normalized and averaged across all four percentages and three temporal conditions (12 conditions overall), as their patterns were comparable. Top: random dynamic OSNs, starting from RE network. Bottom left: BA dynamic OSNs. Bottom right: DM dynamic OSNs. The same color bar is used in all three plots, ranging from the average performance of random alignment (comparable in all three datasets) to perfect alignment (1.0).

Some natural directions include PATENet as an OSN kernel, to use in classification and regression problems where the input to the classifier is an OSN. Possible applications include assigning subjects to different categories (e.g. normal development, accelerated development, retarded development) based on the observed development from longitudinal studies. Another natural direction for future work is to extend PATENet to align multiple OSNs (as opposed to a pair of OSNs). The resulting multi-sequence variant of PATENet can also be used to cluster OSNs.

5.2 Generalizations

The empirically-demonstrated version of PATENet is limited to the case where the elements of the OSNs are undirected networks with pre-specified correspondence between nodes in each element of one OSN and nodes in each element of any other OSN to be aligned with it. It would be interesting to explore variants of PATENet that can work with OSNs consisting of directed graphs, graphs with both directed and undirected edges, or colored graphs (with multiple types of nodes and/or edges), etc. It would also be interesting to consider variants of PATENet that can work in settings where the correspondence between nodes in each element of one OSN and nodes in each element of any other OSN to be aligned with it is not specified, but instead needs to be established based on some node similarity criteria [18].

Furthermore, while in this paper we have focused on the pairwise alignment of OSNs, the PATENet algorithm can be further generalized to work with ordered sequences of arbitrary elements (instead of networks) so long as we can specify a well-behaved unsigned normalized similarity measure between such elements.

6 Conclusion

Networks that change over time, e.g. functional brain networks that change their structure due to processes such as development or aging, are naturally modeled by TENs. Longitudinal measurements of such TENs are naturally represented as OSNs, where each network in the sequence represents a static snapshot of the TEN at a specific time of observation. In this paper we proposed PATENet, a novel algorithm for optimal local alignment of a pair of OSNs. The algorithm requires three user-defined inputs in addition to a pair of OSNs to be aligned: a well-defined UNSSM, a signed normalized monotonically increasing transform, and a match threshold. We showed how PATENet can be used to compute an alignment score, as well as a similarity score, for a pair of aligned OSNs.

Our experiments using PATENet to align synthetic OSNs produced using different generative models of OSNs with their noise corrupted counterparts show that: at lower match thresholds, PATENet displays a high degree of noise tolerance, significantly outperforming random alignment over a broad range of noise levels; at higher match thresholds (more stringent match criteria), PATENet shows increased susceptibility to noise.

PATENet offers a mathematically sound approach to aligning OSNs, which is amenable to being generalized along a number of dimensions, e.g. OSNs consisting of directed networks, labeled networks, or even ordered sequences of other types of elements.

Acknowledgments. This project was supported in part by the National Center for Advancing Translational Sciences, National Institutes of Health through Grant UL1 TR000127 and TR002014, by the National Science Foundation, through Grant SHF 1518732, the Center for Big Data Analytics and Discovery Informatics at Pennsylvania State University, the Edward Frymoyer Endowed Professorship in Information Sciences and Technology at Pennsylvania State University and the Sudha Murty Distinguished Visiting Chair in Neurocomputing and Data Science funded by the Pratiksha Trust at the Indian Institute of Science [both held by Vasant Honavar]. The content is solely the responsibility of the authors and does not necessarily represent the official views of the sponsors. We thank Sanghack Lee for helpful discussions during the course of this work.

References

1. Albert, R., Barabasi, A.L.: Topology of evolving networks: local events and universality. Phys. Rev. Lett. **85**(24), 5234–5237 (2000). https://doi.org/10.1103/PhysRevLett.85.5234
2. Barabasi, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999). https://doi.org/10.1126/science.286.5439.509
3. Bassett, D.S., Sporns, O.: Network neuroscience. Nat. Neurosci. **20**(3), 353–364 (2017). https://doi.org/10.1038/nn.4502
4. Caspi, Y., Irani, M.: Spatio-temporal alignment of sequences. IEEE Trans. Pat. Anal. Mach. Int. **24**(11), 1409–1424 (2002). https://doi.org/10.1109/TPAMI.2002.1046148
5. Dorogovtsev, S.N., Mendes, J.F.F.: Scaling behaviour of developing and decaying networks. Europhys. Lett. **52**(1), 33–39 (2000). https://doi.org/10.1209/epl/i2000-00400-0
6. Elzinga, C.H.: Distance, similarity and sequence comparison. In: Blanchard, P., Bühlmann, F., Gauthier, J.A. (eds.) Advances in Sequence Analysis: Theory, Method, Applications. LCRSP, vol. 2, pp. 51–73. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04969-4_4
7. Emmert-Streib, F., Dehmer, M., Shi, Y.: Fifty years of graph matching, network alignment and network comparison. Inf. Sci. **346**, 180–197 (2016). https://doi.org/10.1016/j.ins.2016.01.074
8. Erdos, P., Renyi, A.: On random graphs I. Publ. Math. Debrecen **6**, 290–297 (1959)
9. Holme, P.: Modern temporal network theory: a colloquium. Eur. Phys. J. B **88**(9), 234–263 (2015). https://doi.org/10.1140/epjb/e2015-60657-4
10. Holme, P., Saramki, J.: Temporal networks. Phys. Rep. **519**(3), 97–125 (2012). https://doi.org/10.1016/j.physrep.2012.03.001
11. Kivela, M., Arenas, A., Barthelemy, M., Gleeson, J.P., Moreno, Y., Porter, M.A.: Multilayer networks. J. Complex Netw. **2**(3), 203–271 (2014). https://doi.org/10.1093/comnet/cnu016
12. Koutra, D., Vogelstein, J.T., Faloutsos, C.: DELTACON: a principled massive-graph similarity function. In: Proceedings of the 2013 SIAM International Conference on Data Mining, pp. 162–170. SIAM (2013). https://doi.org/10.1137/1.9781611972832.18
13. Lee, W.N., Das, A.K.: Local alignment tool for clinical history: temporal semantic search of clinical databases. In: AMIA Annual Symposium Proceedings, pp. 437–441 (2010)
14. Li, A., Cornelius, S.P., Liu, Y.Y., Wang, L., Barabasi, A.L.: The fundamental advantages of temporal networks. Science **358**(6366), 1042–1046 (2017). https://doi.org/10.1126/science.aai7488
15. Luo, G., Cordier, F., Seo, H.: Spatio-temporal segmentation for the similarity measurement of deforming meshes. Vis. Comput. **32**(2), 243–256 (2016). https://doi.org/10.1007/s00371-015-1178-8
16. Madhyastha, T., Peverill, M., Koh, N., McCabe, C., Flournoy, J., Mills, K., King, K., Pfeifer, J., McLaughlin, K.A.: Current methods and limitations for longitudinal fMRI analysis across development. Dev. Cogn. Neurosci. (2017). https://doi.org/10.1016/j.dcn.2017.11.006
17. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. J. Mol. Biol. **147**(1), 195–197 (1981). https://doi.org/10.1016/0022-2836(81)90087-5
18. Towfic, F., Greenlee, M.H.W., Honavar, V.: Aligning biomolecular networks using modular graph kernels. In: Salzberg, S.L., Warnow, T. (eds.) WABI 2009. LNCS, vol. 5724, pp. 345–361. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04241-6_29

Automated Identification of Potential Conflict-of-Interest in Biomedical Articles Using Hybrid Deep Neural Network

Incheol Kim(✉) and George R. Thoma

Lister Hill National Center for Biomedical Communications,
National Library of Medicine, 8600 Rockville Pike, Bethesda, MD 20894, USA
{ickim, gthoma}@mail.nih.gov

Abstract. Conflicts-of-interest (COI) in biomedical research may cause ethical risks, including pro-industry conclusions, restrictions on the behavior of investigators, and the use of biased study designs. To ensure the impartiality and objectivity in research, many journal publishers require authors to provide a COI statement within the body text of their articles at the time of peer-review and publication. However, author's self-reported COI disclosure often does not explicitly appear in their article, and may not be very accurate or reliable. In this study, we present a two-stage machine learning scheme using a hybrid deep learning neural network (HDNN) that combines a multi-channel convolutional neural network (CNN) and a feed-forward neural network (FNN), to automatically identify a potential COI in online biomedical articles. HDNN is designed to simultaneously learn a syntactic and semantic representation of text, relationships between neighboring words in a sentence, and handcrafted input features, and achieves a better performance overall (accuracy exceeding 96.8%) than other classifiers such as support vector machine (SVM), single/multi-channel CNNs, Long Short-term Memory (LSTM), and an Ensemble model in a series of classification experiments.

Keywords: Conflict-of-interest · Two-stage machine learning
Hybrid deep neural network · MEDLINE®

1 Introduction

Conflict of interest (COI) is defined as a situation where a primary interest will be compromised or unduly influenced by a secondary interest. From the biomedical field point of view, primary interests represent health of patients, integrity of research, or duties of public office. A secondary interest generally includes a financial gain for the author (or author's spouse or dependents) received from, or personal relationship with, individuals or "for-profit" organizations such as pharmaceutical companies. Financial conflict-of-interest (FCOI) in biomedical research may cause a number of potential ethical risks, including an increased possibility of pro-industry conclusions, restrictions on the behavior of the investigators, and the use of biased study designs.

MEDLINE®, the U.S. National Library of Medicine (NLM)'s premier online bibliographic database containing more than 25 million citations and abstracts from over 5,600 biomedical journals published in the United States and in other countries, recently announced that it will add COI information to article abstracts available through PubMed [1] when COI declaration statements are supplied by the publishers, to allow users to judge the credibility of findings in published articles. Many biomedical journal publishers also require authors to provide a COI statement within the body text of their articles at the time of peer-review and publication, thereby letting reviewers and readers easily know the integrity of research. However, author's self-reported COI disclosure often does not explicitly appear in their article, and may not be very accurate or reliable due to the lack of author's understanding of relatedness between a certain financial gain they received and their current research. Moreover, there have been no means or systems to verify the accuracy of such authors' COI disclosure.

In this paper, we present an automated method for identifying a potential COI from online biomedical articles using a deep learning-based text classification technique. Our idea is to identify a sentence called COI sentence that contains information of funding support from "for-profit" organizations from the body text of a given biomedical article. This task is quite challenging due to the wide range of linguistic expressions and writing styles, and especially similar expressions for "non-profit" funding sources and a personal acknowledgment.

In order to tackle such challenges, we designed and developed a two-stage machine learning scheme. In stage 1, we distinguish all "support" sentences containing information on any financial support authors received for their research from the body text of an article. In stage 2, these "support" sentences are then classified into two classes according to their funding sources: "for-profit" and "non-profit". Our two-stage machine learning scheme is implemented using a hybrid deep neural network (HDNN) built on combining a multi-channel convolutional neural network (CNN) and a feed-forward neural network (FNN). The CNN component in the proposed HDNN is responsible for learning a syntactic and semantic representation of a text, and contextual relationships between neighboring words in a sentence, while the FNN section takes care of handcrafted input features.

We evaluated the proposed HDNN by comparing its classification performance with that of other types of classifiers such as support vector machine (SVM) with a radial basis kernel function (RBF), single/multi-channel CNNs, Long Short-term Memory (LSTM), voting scheme, and Ensemble model. Three types of word vectors (embeddings): two dense and distributed representations known as Word2Vec [2] and GloVe [3] and a dictionary-based sparse and discrete representation of words, are employed to convert an input sentence into two-dimensional input vector representation and to build an embedding layer for the CNNs. In addition, a bag of words (BOW) based on unigram word statistics representing how differently a word is distributed in "support" and other sentence classes is also used as an input feature for the SVM and the FNN section in HDNN.

2 Related Works

Identifying a sentence that suggests “for-profit” or “non-profit” funding support to determine a potential COI belongs to a text classification or categorization task, a popular topic in the field of natural language processing (NLP). Automated text classification is the process of automatically assigning one or more of a set of predefined categories to a given text or document based on its content, and has been addressed by various methods based on statistical theories and machine learning techniques such as Naïve Bayes [4], decision tree [5], and SVMs [6]. In recent years, deep learning techniques [7] have set a new breakthrough trend in machine learning due to the remarkable success in tackling complex learning problems, and ease of access to high performance computing resources and state-of-the-art open source libraries.

CNN has emerged as the most commonly and widely used architecture in deep learning. It was originally developed for computer vision-related tasks but has been also shown to be very effective for various text classification and understanding tasks such as sentiment analysis and question-answering [8–10]. CNN can learn syntactic and semantic representations of a text, and capture relationships between neighboring words in a sentence automatically through convolution and pooling operations for a sequence of 1-dimensional word or character embeddings. A simple CNN architecture achieves very strong results [11], and can therefore serve as a drop-in replacement for the abovementioned conventional machine learning methods.

Recurrent neural network (RNN) is another popular deep learning architecture especially for NLP tasks. Unlike CNN where the architecture is hierarchical and zero padding or rip out is required to make a fixed-sized word (or character) sequence, RNN is sequential and able to naturally handle word sequences of any length. Other variants such as LSTM [12, 13] were also designed to avoid the problem of exploding or vanishing gradients in the standard RNN and to better capture long-term dependencies.

More recently, an ensemble approach [14] which combines different types of multiple pre-trained classifiers has been proposed to achieve better performance by compensating for errors from individual classifiers. Our proposed HDNN also combines multiple neural network (NN) architectures: a multi-channel CNN section designed to employ different types of word embeddings, and a conventional FNN section for high-performing handcrafted input features. However, our approach totally differs in that each individual NN section in the HDNN is not pre-trained and tightly combined during a learning phase, through the full connection between hidden layers and output layer. We demonstrate the effectiveness of our method by comparing it with the other abovementioned deep neural models, as well as other conventional machine learning techniques such as SVM and voting scheme, with respect to their accuracy in identifying potential COI information from biomedical articles.

3 Conflict-of-Interest: Issues and Challenges

There is an increased concern about the impact of financial relationships between biomedical researchers or their institutions and the pharmaceutical industry on the integrity of biomedical research. Thus, the National Institutes of Health (NIH), as the nation's biomedical research agency, has strict regulations regarding FCOI to ensure impartiality and objectivity in the research it funds [15]. According to NIH regulations, FCOI may exist if investigators or their spouse and dependents received financial support such as "consulting fee", "honoraria", "travel cost", and "royalty" from the third-party private companies. Financial support from "non-profit" organizations such as a local or federal government agency is not considered as a potential COI. Typical examples of author's self-declared COI statements are shown in Table 1.

Table 1. Examples of author's self-declared COI statements.

Supports	Author's self-declared COI statements
Consulting fee	Dr. Hodi reports receiving consulting fees from Bristol-Myers Squibb-Medarex, Novartis, and Genentech; Dr. O'Day, receiving consulting fees, grants, honoraria, and fees for participation in speakers' bureaus from Bristol-Myers Squibb.
Patent	Dr. NL Saccone is the spouse of Dr. SF Saccone, who is also listed as an inventor on the above patent.
Stock	Diane Warden, Ph.D., M.B.A. has owned stock in Pfizer, Inc. within the last five years.
Royalty & travel cost	JAS receives licensing royalties from Genzyme/Sanofi for eliglustat tartrate (Cerdelga) and related compounds. BES has received travel support from Shire HGT and Genzyme.

Table 2. Examples of author's COI disclosure (bold and italicized) in the acknowledgment section in biomedical articles.

We thank the staff at the Metabolic Research Unit at the Jean Mayer USDA Human Nutrition Research Center on Aging at Tufts University, and at the Children's Nutrition Research Center, Department of Pediatrics, Baylor College of Medicine for their work on the study. LC was supported by grant DK007651. ***This research was supported by the Unilever Corporate Research, Bedfordshire, UK***.
The study was supported by NIMH R01MH070437 and K24MH075867. Dr. Druss had full access to all of the data in the study and takes responsibility for the integrity of the data and the accuracy of the data analysis***. Dr. Druss has received funding in the past from Pfizer for a research study.***
This project was funded by the American Academy of Child and Adolescent Psychiatry Physician Scientist Program in Substance Abuse K12 Award (DA 000357-06AK12) and National Institute on Drug Abuse grants U10 DA013732, DA012845 and 5R01DA022284. ***Medication and matching placebo were supplied by Eli Lilly.***
The authors thank the Sansum Diabetes Research Center and the University of California, Santa Barbara, which provided the Artificial Pancreas Software developed by Dr. Eyal Dassau and colleagues***. Product support from both Insulet, for the OmniPod Insulin Management System, and DexCom, for the STS-Seven System and sensors, is acknowledged.***

Currently, biomedical journal publishers rely on an author's self-reported disclosure in determining the existence of potential COI. However, such a disclosure may not be very accurate or reliable due to the lack of author's understanding of relationship between a certain financial gain they received and their current research. Authors also often do not provide a separate section or paragraph for the explicit COI disclosure in their article. Instead, COI statements or sentences appear implicitly and in a subtle manner at the end of the body text or within the acknowledgment, footnote, or appendix section along with other similar sentences that acknowledge a personal support or a funding support received from the "non-profit" organizations as shown in Table 2. Moreover, the wide variety of author's linguistic expressions and writing styles makes the problem of identifying COI information even more challenging. Furthermore, identifying COI manually is time-consuming and labor intensive. To our knowledge, there are no automated systems to verify the accuracy of such authors' COI disclosure.

4 Proposed Method

Our strategy is to identify COI sentences containing a "for-profit" funding support by adopting a two-stage machine learning scheme as shown in Fig. 1. First, an input text which is usually the body text of a given article is divided into sentences in the preprocessing step. Next, all "support" sentences containing information of any financial supports authors received for their research are determined in the first classification stage. Finally, these "support" sentences are classified into two classes according to their funding sources: "for-profit" or "non-profit" in the second stage.

Actually, this task could have been considered a three-class classification problem as sentences belonging to (1) "for-profit", (2) "non-profit", or (3) Others. However, support sentences, especially "for-profit" support sentences, are much rarer compared to "other" sentences within the body text of biomedical articles, thereby heavily skewing the distribution of sentences in each class. Thus, we choose to employ a machine learning scheme having two separate and sequential classification stages. Each stage of classification is performed using a HDNN model that combines a multichannel CNN and a conventional FNN.

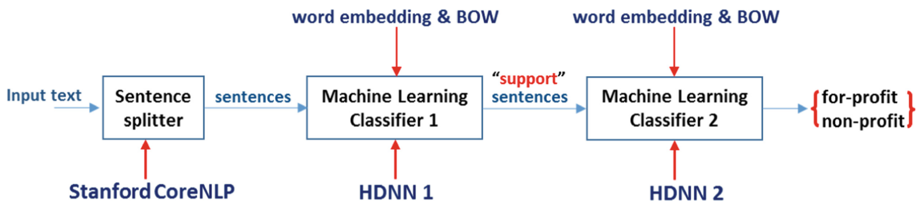


Fig. 1. The proposed two-stage machine learning scheme to identify potential COI.

4.1 Preprocessing: Sentence Splitting

Splitting the text in the body of a biomedical article into individual sentences is an important preprocessing step for the next classification step. Generally, the text in Acknowledgment, Footnote, and Note sections where a "support" sentence is most frequently located, are found to have a more complicated structure than those in the body text. This can be seen from the examples in Table 3, where acronyms or abbreviations for author names, organizations, and degree titles are commonly found in the text. Moreover, the text in these sections often consists of irregular or incomplete sentences. Therefore, it would be not easy to extract sentences using some simple rules based on delimiters such as punctuation marks. In order to deal with this problem, we employed Stanford CoreNLP [16], which is a widely used integrated NLP toolkit including Part-of-speech (POS) tagger, Named entity recognizer (NER), Dependency parser, etc. Its built-in tokenizer has the ability to efficiently and rapidly split sentences.

Table 3. Examples of text containing acronyms or abbreviations for author names, organizations, and degree titles.

Disclosure Summary: J.J.G., K.C., K.A.S., S.B.S.K., E.V.R., and J.M.Z. have nothing to declare. S.M.W.R. consults for Vanda Pharmaceuticals, Inc., through Monash University. C.A.C. has received consulting fees from or served as a paid member of scientific advisory boards for Cephalon, Inc.; Eli Lilly and Co.; Johnson & Johnson; Koninklijke Philips Electronics, N.V./Philips Respironics, Inc.; Sanofi-Aventis Groupe; Sepracor, Inc.; Somnus Therapeutics, Inc.; and Zeo, Inc.
The authors thank the study subjects and their parents for participating in these studies, and the following contributors: Study 1 (M06-888), Edward A. Cherlin, M.D. of Valley Clinical Research, Inc., Andrea Corsino, R.N., M.S.N. of Consultants in Neurology, Ltd., Judith C. Fallon, M.D. of NeuroScience, Inc., David G. Krefetz, D.O., M.D. of CRI Worldwide, Alan J. Levine, M.D. of Alpine Clinical Research Center. Statistical experts were Weining Z. Robieson, Ph.D. (M06-888) and Coleen M. Hall, M.S. (M10-345), of Abbott.

4.2 Input Vector Representation

Our proposed HDNN accepts two different types of input vector representations for a given input sentence: (1) a sequence of word embeddings for the multi-channel CNN section and (2) bag of words (BOW) for the FNN section. These input vector representations are also employed in other types of machine learning models implemented and tested in our study for comparison.

Word Embeddings. In order to perform a text classification task or natural language processing at large using CNN (or other deep neural models such as LSTM), we first need to convert an input sentence or a document to an $n \times k$ matrix. Each row in the matrix is k-dimensional vector representation called word embedding for each individual word in the sentence of length (number of words) n. In our study, we define n as the maximum number of words in a sentence we can find from our training dataset and pad a sentence of length m ($<n$) with $n - m$ of zeros to make the same size of input matrix for CNN.

We employ three word embedding methods: (1) dictionary-based (look-up table), (2) Word2Vec [2], and (3) Glove [3]. Dictionary-based embedding is a sparse and discrete vector representation of word. A dictionary is created using vocabulary words collected from the training dataset. A word vector represents the index of its corresponding word in the dictionary. Word2Vec is a "prediction-based" unsupervised (more precisely, self-supervised) neural network language model. Unlike a dictionary-based word vector, it generates a dense and distributed numerical vector representation of a word. The basic idea of using Word2Vec is to map semantically similar words or words having similar context to nearby points in a lower dimensional vector space. We actually use the publicly available Word2Vec model pre-trained on 100 billion words from Google News. Lastly, we also use another pre-trained model, GloVe, a new global log-bilinear regression model trained on 840 billion tokens of word data for unsupervised learning of global word-word co-occurrence statistics.

All of the three embedding methods generate a 300-dimensional vector representation for each word in a given sentence. Words not present in the dictionary or pre-trained model are represented by an all-zero (in dictionary-based) or a randomly and uniformly initialized (in Word2Vec and GloVe) vector. Although other approaches exploiting character-level embeddings [17] have been also reported, we mainly focus on these word-level embedding methods in this study.

Bag of Words (BOW). We adopt a bag of words (BOW) based on word statistics representing how differently a word is distributed in "support" and "others" sentence classes, to build an input feature vector for the FNN section in HDNN. Word selection to build a BOW is accomplished by sorting words according to their importance measured by simplified $\chi^2(s\chi^2)$ statistics [18].

In our task, $s\chi^2$ of word t_k for sentences in the "support" class (class c_0) and those in the "others" class (class c_1) can be defined as follows:

$$s\chi^2(t_k, c_i) = P(t_k, c_i) \cdot P(\bar{t}_k, \bar{c}_i) - P(t_k, \bar{c}_i) \cdot P(\bar{t}_k, c_i) \quad i = 0, 1 \tag{1}$$

where $P(t_k, c_i)$ denotes the probability that, for a random sentence x, word t_k occurs in x, x belongs to class c_i, and is estimated by counting its occurrences in the training set. The importance of word t_k is finally measured as follows:

$$s\chi^2_{max}(t_k) = max_i s\chi^2(t_k, c_i) \quad i = 0, 1 \tag{2}$$

Accordingly, the more differently a word is distributed in "support" and "others" classes, the higher its $s\chi^2_{max}(t_k)$. Words are sorted according to their $s\chi^2_{max}$, and a BOW is then created by selecting words having highest $s\chi^2_{max}$ scores. Through a series of experiments to investigate the influence of word reduction, we discovered that this BOW feature shows the best performance when its word dictionary size is 500. Finally, the BOW is converted to a binary vector: the vector dimension corresponds to the number of words (=500) in the dictionary, and each vector component is assigned 1 if the corresponding word in the dictionary is found in a given sentence or 0 otherwise. Another BOW for “for-profit” and “non-profit” sentence classes is also obtained through the same procedure above.

4.3 HDNN Architecture

Our proposed HDNN consists of two neural network sections: multi-channel CNN and FNN sections, as illustrated in Fig. 2. Each section takes different types of input vector representations for a given input sentence and then proceeds with the learning process separately until they are combined through a full connection between the hidden layers and output layer. As a result, a hybrid structure is created.

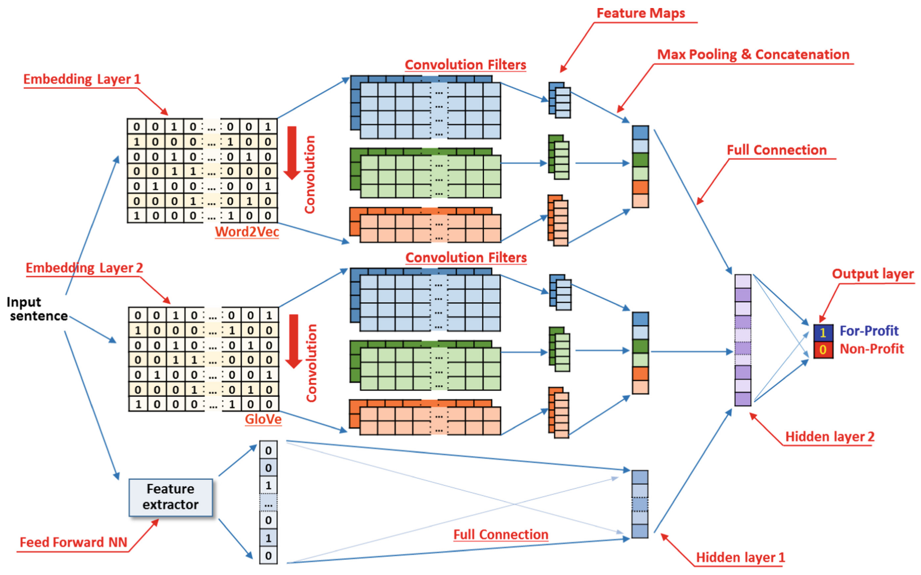


Fig. 2. The structure of proposed HDNN.

Multi-channel CNN Section. The multi-channel CNN section in our HDNN is built on combining two single-channel CNNs. These single-channel CNNs both have the same structures; the size of the matrix word embeddings, the number and size of convolution filters, and other hyperparameters are all identical. However, they employs different types of word embeddings as an input: Word2Vec and GloVe, respectively.

In each CNN, we first apply a convolution filter, $w \in \mathbb{R}^{d \times k}$ and a nonlinear function f with a bias term $b \in \mathbb{R}$ on a window of d rows in the matrix word embeddings, $X \in \mathbb{R}^{n \times k}$, where i^{th} row is the k-dimensional word embedding x_i representing i^{th} word in a given input sentence of length n, as follows;

$$c = f(w \cdot X_d + b) \tag{3}$$

By repeatedly performing such convolution operation over the entire word embedding matrix with stride 1, we can then obtain a feature map $\boldsymbol{c} = [c_1, c_2, \ldots, c_{n-d+1}]$ with $\boldsymbol{c} \in \mathbb{R}^{n-d+1}$. Since we employ the multiple convolution filters (=*h*) with different width or windows sizes (=*l*) for our CNN, we finally have a set of feature maps $\boldsymbol{C} = \left[\boldsymbol{c^1}, \boldsymbol{c^2}, \ldots, \boldsymbol{c^{h \times l}}\right]$.

Next, max-pooling operation is performed to obtain the maximum value from each feature map;

$$y^j = max_i c_i^j \tag{4}$$

where $j = 1, 2, \ldots, h \times l$ and $i = 1, 2, \ldots, n - d + 1$. Such max-pooling operation not only reduces the output dimensionality while keeping the most salient information but also induces a fixed-length of feature vector from the different size of feature maps resulting from applying a different width of convolution filters. These maximum values called features generated from each CNN are concatenated together to form a multi-channel structure, and along with the outputs from the hidden layer in the FNN section, fed to the fully connected next hidden layer. All CNN models implemented in our experiments have 128 convolution filters with three different window sizes (*l* = 3, 4, and 5) generating a total of 384 feature maps, dropout rate of 0.5, mini-batch size of 64, and "Adam" optimizer. All these hyperparameters were obtained through a grid search method. In addition, rectified linear unit (ReLU) and softmax nonlinear activation functions are applied to the hidden layers and the final output layer, respectively.

On the other hand, a similar concept of two-channel approach has been recently suggested by Kim [11]. Unlike our multi-channel approach where each channel of CNN accepts a different type of word embedding and performs a separate convolution operation, his method employs Word2Vec only as an input representation for both channels of CNN; Word2Vec in one channel is kept unchanged (static) and the other is fine-tuned (non-static) during a learning phase.

FNN Section. We introduce the FNN section into HDNN to take advantage of a handcrafted input feature experimentally found to be effective. As mentioned previously, we employ a 500-dimension binary vector representing a BOW as an input feature vector for the FNN section. Words in this BOW are selected and sorted according to their corresponding $s\chi^2_{max}$ scores that reflect the difference of their statistical distributions between the "support" and "others", or "for-profit" and "non-profit" classes.

5 Classification Experiments

5.1 Ground-Truth Dataset and Tools

In order to build a ground-truth dataset for our experiments, we first downloaded 2,800 HTML-formatted online biomedical articles having citation information of grant support or ClinicalTrials.gov from NLM's PubMed Central (PMC) [19]. These articles were published in 938 different journals and indexed in MEDLINE between 2010 and 2015. We then collected a total of 21,822 sentences from these articles and divided them into two classes: "support" and "others" according to whether they contain information of a funding support or not. Sentences in the "support" class were further divided into two sub-classes: "for-profit" and "non-profit".

Among these, 16,753 sentences consisting of 4,528 in the "support" class and 12,225 in the "others" class were randomly selected to train the classifiers for the stage 1 classification experiment—distinguishing "support" sentences from the body text of biomedical articles. The remaining 5,069 sentences (1,509 from the "support" class + 3,560 from the "others" class) were used as a test set to evaluate the performance of our classifiers. Next, for the stage 2 classification experiment—classifying a "support" sentence into "for-profit" or "non-profit" class, we reemployed the "support" sentences already used for the stage 1 classification experiment. Accordingly, each training and testing set has 4,528 (1,937 from the "for-profit" + 2,591 from the "non-profit") and 1,509 (645 from the "for-profit" + 864 from the "non-profit") "support" sentences, respectively.

All DNN models employed in our study including the proposed HDNN, single/multi-channel CNNs, and LSTM were implemented based on Tensorflow [20], very well-known open source library developed by the Google Brain team, Keras [21], a simple and high-level model definition interface, and Nvidia's CUDA toolkit and CuDnn for GPU-acceleration. In addition, SVM with RBF kernel function, another classifier widely used in text classification and other machine learning tasks, was implemented using LibSVM [22], a free software package.

5.2 Experimental Results

As mentioned earlier, our proposed method of identifying potential COI from an online biomedical article adopts a two-stage machine learning scheme. In experiments, we implemented and evaluated a total of 9 classifiers for both stage 1 and 2 classification tasks: SVM with a RBF, 3 single-channel CNNs with different word embeddings, LSTM, multi (two)-channel CNN, voting scheme, Ensemble model, and our proposed HDNN.

First, it can be clearly seen from Tables 4 and 5 that all DNN models consistently outperform SVM. In the case of LSTM, a better performance was achieved than that of any of single-channel CNNs in the stage 1 classification experiments. However, a reversal in performance is observed in the stage 2 classification. Note that the size of the training dataset used in the stage 2 experiments is significantly smaller (about 25%) than that in the stage 1 experiments. Thus, LSTM is analyzed to be more susceptible to the size of the training dataset than other classifiers, thereby resulting in a degradation of the classification performance in stage 2.

We can also see that our multi-channel CNN and especially HDNN both accepting multiple input representations: "word2Vec + GloVe" and "Word2Vec + GloVe + BOW", respectively, yield the best performance overall in both stage 1 and 2 classification experiments. The ensemble model which also employs three types of input representations the same as those used in the proposed HDNN, through the combination

Table 4. Stage 1 classification results.

Models	Accuracy	Precision	Recall	F_1
SVM	96.21	98.59	95.98	97.27
Dic CNN	96.61	96.67	98.57	97.61
W2v CNN	97.04	97.28	98.54	97.91
Glv CNN	97.18	98.03	97.95	97.99
W2v+Glv CNN	97.93	98.40	98.65	98.53
LSTM	97.57	98.02	98.54	98.28
HDNN	**98.11**	**98.52**	**98.79**	**98.65**
Ensemble	97.40	97.69	98.62	98.15
voting	97.73	97.97	98.82	98.39

Table 5. Stage 2 classification results.

Models	Accuracy	Precision	Recall	F_1
SVM	95.16	94.89	96.76	95.82
Dic CNN	96.02	95.79	97.34	96.56
W2v CNN	96.16	95.90	97.45	96.67
Glv CNN	96.22	96.11	97.34	96.72
W2v+Glv CNN	96.62	96.67	97.45	97.06
LSTM	95.89	95.67	97.22	96.44
HDNN	**96.89**	**97.44**	**97.11**	**97.28**
Ensemble	96.49	96.56	97.34	96.95
voting	96.49	96.23	97.69	96.96

of pre-trained 2 CNNs and FNN, is found to provide a slight improvement over the individual classifiers having a single input vector representation such as SVM, single channel CNNs, and LSTM, but to be not as good as HDNN. Rather, a simple majority voting scheme for the outputs of 5 pre-trained individual classifiers (SVM + 3 CNNs + LSTM) performs better. Therefore, we conclude that our HDNN is a more effective architecture for combining multiple learning models and taking advantage of different types of input representations to boost the overall classification performance further.

Finally, in Table 6 we show some examples of false-negative (FN) and false-positive (FP) errors in stage 2 classification. Here, FN means that “non-profit” support sentence is misclassified into “for-profit” class. FP is the reverse of the above. The first “support” sentence in the FN error examples contains two NIH grant numbers “GM103429” and “GM103450”. However, this sentence is very short, and there is no contextual description associating these grant numbers with an NIH financial support. The second sentence of FN errors is analyzed to be misclassified due to several pairs of words such as “senior advisor”, “pharmaceutical company”, and “international market” which are also frequently found in “for-profit” sentences, even though it has the word, “nonprofit”. In contrast, FP errors shown in Table 6 result from the existence of “non-profit” organizations names (“Cleveland Clinic” and “NIH”) along with a description of “for-profit” funding support within the sentence.

Table 6. Error examples showing FN and FP errors in the stage 2 classification.

Error types	Support sentences
False Negative	JS received financial support from GM103429 and GM103450.
	He was senior clinical advisor of a nonprofit (501c3) pharmaceutical company studying a lower-cost IUD for the U.S. and international markets (Medicines 360).
False Positive	Mass spectrometry studies were performed in the Cleveland Clinic Mass Spectrometry core facility, which is partially supported by a Center of Innovation Award from AB SCIEX.
	The vitamin E softgels and matching placebo were provided by Pharmavite through a Clinical Trial Agreement with the NIH.

6 Conclusions

Conflicts of interest have a major negative impact on the integrity of biomedical research. Many biomedical journal publishers thus require authors to provide a COI disclosure statement in their article at the time of peer-review and publication. However, authors often declares a COI implicitly and in a subtle manner. In addition, there are also rising concerns about the accuracy and reliability of author’s self-declared COIs.

In this paper, we have presented a sequential two-stage machine learning-based text classification scheme to automatically ascertain potential COI from the body text of online biomedical articles. The first stage of classification is for distinguishing “support” sentences from other sentences in the body text of a given biomedical article, and the second stage is for categorizing those “support” sentences into “for-profit” and “non-profit” classes according to their funding sources. Each stage of classification is implemented using a deep learning model that has a hybrid architecture to combine a multi-channel convolutional neural network (CNN) and a feed-forward neural network

(FNN). This hybrid deep neural network (HDNN) aims at simultaneously learning syntactic and semantic representations and word relationships in a sentence, and handcrafted input features.

Experiments on a total of 21,822 sentences from 2,800 HTML-formatted online biomedical articles published in 938 different journals show that our proposed HDNN yields a consistently higher performance in both stage 1 and 2 classification tasks, compare to other classifiers having a single type of input vector representation such as SVM, single-channel CNNs, and LSTM. Our HDNN is also found to be a superior architecture for combining multiple input representations than an ensemble model or voting scheme both based on pre-trained learning models.

Acknowledgment. This research was supported by the Intramural Research Program of the National Library of Medicine, National Institutes of Health.

References

1. http://www.ncbi.nlm.nih.gov/pubmed
2. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Proceedings of Advances in Neural Information Processing Systems (NIPS 2013), pp. 3111–3119, Lake Tahoe (2013)
3. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), pp. 1532–1543, Doha, Qatar (2014)
4. Ting, S.L., Ip, W.H., Tsang, A.H.C.: Is Naïve Bayes a good classifier for document classification? Int. J. Softw. Eng. Appl. **5**(3), 37–46 (2011)
5. Mercer, R.E., Di Marco, C.: A design methodology for a biomedical literature indexing tool using the rhetoric of science. In: Proceedings of the HLT-NAACL 2004 Workshop: BioLINK 2004, Linking Biological Literature, Ontologies and Databases, pp. 77–84, Boston (2004)
6. Athar, A.: Sentiment analysis of citations using sentence-structure-based features. In: Proceedings of 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011), pp. 81–87, Portland (2011)
7. LeCun, Y., Bengio, Y., Hinton, G.E.: Deep learning. Nature **521**(7553), 436–444 (2015)
8. Kalchbrenne, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL), pp. 655–665, Baltimore (2014)
9. dos Santos, C.N., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014): Technical Papers, pp. 69–78, Dublin, Ireland (2014)
10. Dong, L., Wei, F., Zhou, M., Xu, K.: Question answering over freebase with multi-column convolutional neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL), pp. 260–269, Beijing, China (2015)
11. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), pp. 1746–1751, Doha, Qatar (2014)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)

13. Sutskever, I., Vinyals, O., Le, Q.: Sequence to sequence learning with neural networks. In: Proceedings of Advances in Neural Information Processing Systems (NIPS 2014), pp. 3104–3112, Montreal, Canada (2014)
14. Ghosal, D., Bhatnagar, S., Akhtar, M.S., Ekbal, A., Bhattacharyya, P.: IITP at SemEval-2017 task 5: an ensemble of deep learning and feature based models for financial sentiment analysis. In: Proceedings of the 11th International Workshop on Semantic Evaluations, pp. 899–903, Vancouver, Canada (2017)
15. https://grants.nih.gov/grants/policy/coi/tutorial2011/fcoi.htm
16. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55–60, Baltimore (2014)
17. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Proceedings of Advances in Neural Information Processing Systems (NIPS 2015), pp. 649–657, Montreal, Canada (2015)
18. Galavotti, L., Sebastiani, F., Simi, M.: Experiments on the use of feature selection and negative evidence in automated text categorization. In: Borbinha, J., Baker, T. (eds.) ECDL 2000. LNCS, vol. 1923, pp. 59–68. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45268-0_6
19. http://www.ncbi.nlm.nih.gov/pmc/
20. Abadi, M., Agarwal, A. et al.: Tensorflow: large-scale machine learning on heterogeneous distributed systems. Software (2015). tensorflow.org
21. Chollet, F., et al.: Keras. GitHub (2015). https://github.com/fchollet/keras
22. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001). http://www.csie.ntu.edu.tw/∼cjlin/libsvm

Hierarchical Bayesian Classifier Combination

Mohammad Ghasemi Hamed(✉) and Ahmad Akbari(✉)

Audio & Speech Processing Lab, Computer Engineering Department, Iran University of Science and Technology, Tehran, Iran
mohammad.gh@gmail.com, akbari@iust.ac.ir

Abstract. This paper proposes a Bayesian method for combining the output of multiple base classifiers. The focus is put on combination methods for merging the outputs of several and possibly heterogeneous classifiers with the aim of gaining in the final accuracy. Our work is based on the Dawid and Skene's work [11] for modelling disagreement among human assessors. We also take advantage of the Bayesian Model Averaging (BMA) framework without requiring the ensemble of base classifiers to correspond in a mutually exclusive and exhaustive way to all the possible data generating models. This makes our method relevant for combining multiple classifiers' output each observing and predicting the behavior of an entity by means of divers aspects of the underlying environment. The proposed method, called Hierarchical Bayesian Classifier Combination (HBCC) is for discrete classifiers and assumes that the individual classifiers are conditionally independent given the true class label. The comparison of HBCC with majority voting on six benchmark classification data sets shows that it generally outperforms majority voting in the classification accuracy.

1 Introduction

The recent growth in the field of machine learning and pattern recognition provided important advances in the underlying techniques and led to substantial developments of learning algorithms. These developments produced a plethora of prediction techniques with no one being uniformly better than the rest, on all the possible data sets. When faced with a new prediction problem, it is generally quite difficult to choose the "best" model from a list of prediction models with competitive results. In such cases, where one has little prior knowledge of the new problem, an intuitive idea is to combine the output of different prediction models in the hope that this combination would improve the final prediction performance. This is the goal of ensemble learning which has been a subject of research for the two precedent decades [12,25,28,33]. Ensemble learning has shown to provide models with generally higher prediction performance than single classification or regression models [2,22,26].

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 113–125, 2018.
https://doi.org/10.1007/978-3-319-96136-1_10

Currently, there are several popular techniques for combining classifier such as bagging [6], boosting [16], dagging [31], Stacking [34], Hierarchical Mixture of Expert (HME) [19], Naive Bayes and majority voting [35]. The classification and regression combination techniques can be categorized based on their complexity in two groups: the first one involves bagging, boosting, dagging and other combination techniques combining several models constructed with the same classification or regression method. The motivation of such techniques is to enhance the performance of a restricted prediction model by applying them on modified versions of the same data set. Bagging [6] is a majority vote on models trained independently on the random sample of the training set. Boosting [16] is a linear combination of models sequentially trained on data point sampled with probability proportional to the previous model's error on that point. Dagging [31] is majority voting on models trained independently on disjoint samples of the original training set.

The second group is the collection of ad hoc techniques and meta-learning methods such as Stacking, HME, Naive Bayes and majority voting used to combine different classification or regression models. These techniques could be used for heterogeneous model built by different models. They do not make modifications of the original data set and can also be used upon the ensemble models obtained by the first group. The Hierarchical Mixture of Experts (HME) is a tree-structured architecture for combining the output of several mixture components through weights representing probabilistic splits of the input space [19]. This combination was first suggested with mixture component each being a Generalized Linear Model (GLM) [18] and then extended to other models [36].

Stacking (or "stacked generalization") was first introduced by Wolpert [34] for classification, then it has been generalized to regression [7]. This technique uses a training set composed of the outputs of all ensemble models and the true class label for input instances not in the ensemble models. Then it learns how to combine the individual models in order to minimize the classification or the regression error [14,32].

In this paper, we propose HBCC which assumes that the individual classifiers are conditionally independent given the true class label and employ hierarchical conjugate priors for modelling the combination parameters. The HBCC's combination technique is based on the Dawid and Skene's work [11] for modelling disagreement among human assessors. This work could also be seen as a hierarchichal Bayesian modelling of the Naive Bayes combination approach [35].

This paper is organized as follows: Sect. 2 gives an overview of existing literature on the application of Bayesian methods to ensemble learning. Section 3 introduces HBCC and Sect. 4 reports its application on the benchmark data sets and compares them to majority voting's results.

2 Bayesian Classifier Combination

In Bayesian view, the prediction consists in updating our belief about the new observation according to the observed data and marginalizing the model parameters [3,5]. This turns out in combining the prediction of all possible models in the underlying model space, where each model prediction is weighted by the model's posterior probability. These posterior probabilities reflect the probability of each model being the "true" model or being the Data Generation Model (DGM)[1] and their role in Bayesian prediction is to allow for the uncertainty of the unknown DGM. Such method of combining each models' prediction through a marginalization over the parameter space is known as Bayesian Model Averaging (BMA).

The BMA framework is the optimal Bayesian approach for combining the prediction of multiple models, however it is not suited for Ensemble learning [23]. Several authors [10,13,23], have shown that BMA does not perform well for the task of combining multiple classifier's output. This is justified by the fact that, the motivations and practices of ensemble learning are in contradiction with some fundamental assumptions of BMA.

As stated by [20,23,24], Bayesian model averaging is a well-suited approach to find a classifier among the ensemble classifier which is the most "similar" to the DGM. This is well-motivated when the "true" model that has generated these data, is one of the multiple classifiers. The ensemble of K classifiers must correspond in a mutually exclusive and exhaustive manner to the K possible ways by which the data was generated. All the K classifiers must be trained on the same training set. In fact, we generally know that none of the K model has generated the data, but the idea behind classifier combination is to find a combination of the ensemble classifier which may yield a more accurate classifier than any of the base classifiers.

Moreover, the ensemble classifiers are usually constructed on different data sets. They are usually constructed on sub sample or mutually exclusive sets of the original data set which is a way to reduce the correlation among classifiers. Indeed, each individual classifier estimates the true population distribution, so the output of each classifier is dependent to the true class label. Therefore, the output of the K classifiers are indirectly dependent on each other through their shared dependency to the true class label. Furthermore, the combination may also be used to merge the output of different classifiers constructed based on different sets of variables. Such requirement arises in situations where the true entity of interest lies in a heterogeneous environment that makes possible to observe and predict it by means of various aspects. For example, the opening of new flight line connecting two specific airports, could be predicted by combining the output of several classifier such as: expert opinions, one or many economical models and a classifier based on Internet news and customer comments.

2.1 Bayesian Model Averaging

Although the Bayesian model averaging is an effective and well-motivated method for combining the output of several classifiers, in order to find which of the base classifiers is the optimal one, it yields poor results when the DGM is not one of the models in the ensemble [10]. By empirical comparison, Clark [10] noticed that when the classifier is not in the model, BMA results converges to

[1] The Data Generation Model (DGM) is the model that produced the observed data.

the closest ensemble model to the DGM instead of converging to the closest combination to the DGM.

Ensemble methods are used when single prediction models are not able to grasp the complexity of the data. In such cases, single models may be biased or overfit whereas ensemble methods may find a more robust model by taking advantage of the enriched hypothesis spaces generated through possible combination of component models. In other words, Bayesian model averaging is not intended to exploit the enriched hypothesis space available by the model combination [23].

In full Bayesian, the distribution of the response variable of a model is obtained by the posterior predictive distribution of response variable. The posterior predictive distribution is the probability of the new observation given past data $\mathcal{P}(t^*|x^*, \mathcal{D})$. It is obtained by the integral of the product of the probability of the new observation given the estimated model $\mathcal{P}(t^*|x^*, h)$[2], times the posterior probability of the estimated model given the past data $\mathcal{P}(h|\mathcal{D})$ over all models $h \in \mathcal{H}$.

$$\mathcal{P}(t^*|x^*, \mathcal{D}) = \sum_{h \in \mathcal{H}} \mathcal{P}(t^*|x^*, h, \mathcal{D})\mathcal{P}(h|\mathcal{D}) \tag{1}$$

2.2 Bayesian Methods in Classifier Combination

Xu et al. [35] used the Bayes formula along with the conditional independence assumption on the base classifiers and introduced it as the Naive Bayes approach for classifier combination. Monteith et al. [24] introduced the Bayesian Model Combination (BMC) technique which uses Bayesian inference to generate the optimal weights to combine a set of fixed or already learned classifiers. Their proposed technique is a Bayesian model averaging over the set of all possible linear combination of ensemble classifiers rather than the initial set of ensemble classifiers. The authors proposed two strategies for sampling from the set of all possible linear combinations of ensemble classifiers and they shown that both strategies outperformed bagging, boosting and BMA over a wide variety of cases.

Raykar et al. [27] proposed a Bayesian approach for the case of multiple experts (assessors) providing labels but no absolute gold standard. Their technique, assume assessors to be conditionally independent and jointly estimates the individual assessors, their accuracy and the true (hidden) labels.

Kim and Ghahramani [20] proposed a Bayesian Classifier combination method based on the Haitovsky et al. [17] work being itself a Bayesian extension of Dawid and Skene's work [11] for modelling disagreement among human assessors. Their work [20] proposed the Independent Bayesian Combination Model (IBCC) that models the classifiers as being conditionally independent and use Bayesian inference to learn class prior distribution and classifier confusion matrices. They also introduced three new extensions for modelling the correlations

[2] For a given value of t^* and x^* the distribution $p(t^*|x^*, h)$ depends only on h and remains constant for all values of $\mathcal{D}$. It means that the random variable $t^*|x^*$ is conditionally independent of $\mathcal{D}$ given h which yields $\mathcal{P}(t^*|x^*, h) = \mathcal{P}(t^*|x^*, h, \mathcal{D})$.

between component classifiers. The four proposed methods generally outperformed majority voting on several data sets. IBCC differs from the method proposed by Raykar et al. [27], because the latter gives estimates of the individual assessors and their accuracy whereas, IBCC takes the assessors fixed and infer the confusion matrix which implicitly reflect classifiers accuracy.

Simpson et al. [30] proposed a simpler version of IBCC for classification with variational inference instead of Gibbs Sampling. The proposed method shown greater performance over voting and weighted mean in the experimental setup. It has also the merit of being computationally efficient and make it practical for application demanding regular updates as new data is observed.

Recently, Lacoste et al. [21] introduced Agnostic Bayesian learning of ensembles which is a technique for producing ensembles of predictors based on holdout estimations of their generalization performances. While being efficient and easily adjustable to any loss function, the agnostic Bayes framework, is proposed for addressing model selection problems.

We propose HBCC which is somewhat between the Naive Bayes [35] and the IBCC [20] approaches. HBCC has a hierarchical Bayesian model and its joint distribution is the same as IBCC. However, HBCC assumes, like the Naive Bayes approach, that true class labels are observed in the training set, whereas in IBCC they are modeled by hidden variables. This is an important modelling choice which places HBCC in the category of supervised methods whereas IBCC and the Simpson et al.'s variational inference extension [30] belong to non-supervised methods. This reduction of unobserved random variables yields a generative method simpler to train where the full Bayesian treatment requires a much smaller parameter samples. In other words, HBCC provides classifier combination models having less training and classification time.

It is important to note that the Naive Bayes approach may appear similar to HBCC (or IBCC), because both methods find a separate confusion matrix for each ensemble classifier which is then used to combine their classifier outputs. However, the former takes the confusion matrix as prior data whereas the latter (and IBCC) use Bayesian inference to learn the confusion matrix and the class proportion. Moreover, HBCC (and IBCC) can handle missing data (classifier output) and the confusion matrix is inferred by a hierarchical distribution which allows for more complex models. Besides, the initial Naive Bayes version proposed in [35] works only with probabilistic classifiers, so Xu et al. [35] proposed another version which approximates the term involving class probabilities. The lack of classifier output coverage for all true class label (missing data) is another issue for Naive Bayes combination which is seamlessly handled by BMA using a vague prior.

3 Hierarchical Bayesian Combination Model (HBCC)

3.1 The Classifier Combination Model

In this section we present the Dawid and Skene's method [11] for modelling observer error rate when the true class label is known. This model is presented in [20] where it has been slightly refined for the classifier combination task. The refinement consists in assuming that each classifier produces just one output per observation.

We assume that we have a set of observations $(x_i, t_i), i = (1, \ldots, n)$ where x_i denotes the vector of independent variables and $t_i \in \{1, \ldots, J\}$ denotes the true class label (the response value). This model is composed of a set of classifier $c^k, k \in \{1, \ldots, K\}$ where K is the number of classifiers each proving the output $c_i^k \in \{1, \ldots, J\}$ for the input vector x_i. This combination model assumes classifiers with discrete outputs and does not take into account the vector x_i. Each true class label t_i is supposed to come from a multinomial distribution with parameter $\mathbf{p} = (p_1, \ldots, p_J)$, so we have:

$$\mathcal{P}(t_i = j|\mathbf{p}) = p_j$$

where p_j is the proportion for class j. The classifier output c_i^k is assumed to be generated from a multinomial distribution with parameter $\boldsymbol{\pi}_j^k = (\pi_{j,1}^k, \ldots, \pi_{j,J}^k)$; this is formulated as:

$$\mathcal{P}(c_i^k = l|t_i = j, \boldsymbol{\pi}_j^k) = \pi_{j,l}^k. \tag{2}$$

Note that the collection $\boldsymbol{\pi}^k = \{\boldsymbol{\pi}_1^k, \ldots, \boldsymbol{\pi}_J^k\}$ represents the confusion matrix for classifier k, where each row is the vector of parameters $\boldsymbol{\pi}_j^k$ whose elements are described by Eq. (2). By assuming the classifier outputs c_i^k to be conditionally independent given the true label t_i, the joint probability of t_i and the set of classifiers to give a combination $\mathbf{c}_i = (c_i^1, \ldots, c_i^K)$ is:

$$\mathcal{P}(\mathbf{c}_i, t_i|\mathbf{p}, \boldsymbol{\pi}) = p_{t_i} \prod_{k=1}^{K} \pi_{j,c_i^k}^k,$$

where $\boldsymbol{\pi} = \{\boldsymbol{\pi}^1, \ldots, \boldsymbol{\pi}^K\}$ is the collection of confusion matrices. It is a common practice in classification settings, to assume that the each classifier output is independent and identically generated from their respective distribution and this assumption gets the following likelihood:

$$\mathcal{P}(\mathbf{c}, \mathbf{t}|\mathbf{p}, \boldsymbol{\pi}) = \prod_{i=1}^{n} \left(p_{t_i} \prod_{k=1}^{K} \pi_{j,c_i^k}^k \right), \tag{3}$$

where $\mathbf{t} = (t_1, \ldots, t_n)$ is the vector of all true labels.

3.2 The Hierarchical Bayesian Combination of Classifier Model (HBCC)

Here, we describe the Bayesian treatment for the Sect. 3.1. This model uses hierarchical conjugate priors for confusion matrix and conjugate priors for class proportions. The joint distribution of all variable is the same as the model proposed in [20]. Nevertheless, we assume the true labels t_i are observed in the training set, whereas Kim and Ghahramani [20] considered them as hidden variables.

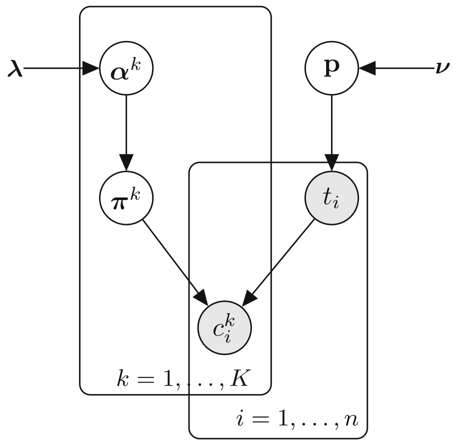


Fig. 1. Graphical model for HBCC. The circular nodes are random variables, the shaded nodes are observed variables and letters $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ represent hyperparameters.

The vector of class proportions $\mathbf{p}$ has a Dirichlet prior distribution with hyperparameters $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_J)$. Each confusion matrix $\boldsymbol{\pi}^k$ is composed of J rows $\boldsymbol{\pi}_j^k = (\pi_{j,1}^k, \ldots, \pi_{j,J}^k)$ Dirichlet distribution with parameters $\boldsymbol{\alpha}_j^k = (\alpha_{j,1}^k, \ldots, \alpha_{j,J}^k)$, and all rows are independent within and across classifiers. Finally, each $\pi_{j,l}^k$ has itself an exponential prior distribution with parameter λ_j^k.

Therefore, $\boldsymbol{\alpha}^k$ represents a matrix having J rows as $\boldsymbol{\alpha}_j^k = (\alpha_{j,1}^k, \ldots, \alpha_{j,J}^k)$, each corresponding to the parameters of the confusion matrix $\boldsymbol{\pi}_j^k$, and the collection of $\boldsymbol{\alpha}^k$ gives $\boldsymbol{\alpha}$. We also have the matrix $\boldsymbol{\lambda}$ being composed of J rows $\boldsymbol{\lambda}_j = (\lambda_{j,1}, \ldots, \lambda_{j,J})$. Note that $\boldsymbol{\lambda}$ is shared among all matrices $\boldsymbol{\alpha}^k$. The described combination model is shown by a directed graphical model in Fig. 1.

The joint posterior distribution $\mathcal{P}(\boldsymbol{\pi}, \mathbf{p}, \boldsymbol{\alpha}, \mathcal{D}|\gamma, \boldsymbol{\nu})$, $\mathcal{D} = \{\mathbf{c}, \mathbf{t}\}$ is defined as:

$$\mathcal{P}(\boldsymbol{\pi}, \mathbf{p}, \boldsymbol{\alpha}, \mathcal{D}|\gamma, \boldsymbol{\nu}) \propto \mathcal{P}(\boldsymbol{\alpha}|\gamma)\mathcal{P}(\boldsymbol{\pi}|\boldsymbol{\alpha})\mathcal{P}(\mathbf{p}|\boldsymbol{\nu})\mathcal{P}(\mathbf{c}, \mathbf{t}|\mathbf{p}, \boldsymbol{\pi}), \tag{4}$$

where $\mathcal{P}(\mathbf{p}|\boldsymbol{\nu})$, $\mathcal{P}(\boldsymbol{\pi}|\boldsymbol{\alpha})$, $\mathcal{P}(\boldsymbol{\alpha}|\gamma)$ and $\mathcal{P}(\mathbf{c}, \mathbf{t}|\mathbf{p}, \boldsymbol{\pi})$ denote respectively the prior Dirichlet distribution of the true label proportions, the prior Dirichlet distribution of the classifiers performance (prior confusion matrix), the prior exponential distribution of confusion matrix parameters and the combination likelihood defined by Eq. (3). Here, we approximate the above posterior distribution via Gibbs sampling. In the next paragraph, we will see that we are interested in sampling from the distribution $\mathcal{P}(\boldsymbol{\pi}, \mathbf{p}|\mathbf{c}_0, \mathbf{t}_0)$ rather than the aforementioned distribution. It goes without saying that sampling from the distribution $\mathcal{P}(\boldsymbol{\pi}, \mathbf{p}|\mathbf{c}_0, \mathbf{t}_0)$ could be achieved by sampling from the distribution described in Eq. (4) and then marginalizing over $\boldsymbol{\alpha}$ (ignoring this variable) and conditioning on $\{\mathbf{c}_0, \mathbf{t}_0\}$ (retaining only observation with $\mathbf{c} = \mathbf{c}_0$ and $\mathbf{t} = \mathbf{t}_0$).

Given a training set $\mathcal{D}_0 = \{\mathbf{c}_0, \mathbf{t}_0\}$ and new observation x^* with classifiers prediction $\mathcal{C}^* = \{c^{*1}, \ldots, c^{*K}\}$, where c^{*k} denotes the output of the k^{th} classifier, the joint posterior distribution on the true class label t^* and classifiers output $\mathcal{C}^*$ conditioned on inferred parameters $\theta_0 = \{\boldsymbol{\pi}_0, \mathbf{p}_0\}$ is defined as:

$$\mathcal{P}\left(t^*, \mathcal{C}^* \middle| \theta_0\right). \tag{5}$$

By averaging (marginalizing) it over the space of possible parameters, $\theta = \{\boldsymbol{\pi}, \mathbf{p}\}$, we obtain the following joint distribution:

$$\mathcal{P}(t^*, \mathcal{C}^* | \mathcal{D}) = \int_{\mathbf{p}} \int_{\boldsymbol{\pi}} \mathcal{P}\left(t^*, \mathcal{C}^* \middle| \theta, \mathcal{D}\right) \overbrace{d\mathcal{P}(\boldsymbol{\pi}|\mathbf{c}, \mathbf{t}) d\mathcal{P}(\mathbf{p}|\mathbf{t})}^{d\mathcal{P}(\theta|\mathcal{D})}. \tag{6}$$

The distribution of $\theta|\mathcal{D}$ is approximated by drawing m observations using Gibbs sampling as described above. Due to the proportionality of posterior and the joint distribution, we have:

$$\underset{t^*}{\operatorname{argmax}}\ \mathcal{P}(t^*|\mathcal{C}^*, \mathcal{D}) = \underset{t^*}{\operatorname{argmax}}\ \mathcal{P}(t^*, \mathcal{C}^*|\mathcal{D}). \tag{7}$$

Therefore, the true label having the maximum posterior probability is obtained by taking the label t^* of the vector $(t^*, \mathcal{C}^*)$ giving the maximum value in Eq. (6). The computational complexity of computing the distribution $\mathcal{P}(t^*, \mathcal{C}^*|\mathcal{D})$ for a new observation by Eq. (6) is $O(mJK)$, where m, J and K are respectively the number of observations obtained from $\mathcal{P}(\theta|\mathcal{D})$, the number of the true class labels and the number of individual classifiers used in the combination.

4 Experiments

In this section we apply the HBCC classifier combination method described in the preceding section to the benchmark data classification data sets listed in Table 1. We compare the method accuracy and computational performance to the majority voting combination technique and their accuracy and computational performances over all (six) data sets are respectively reported in Tables 3 and 2. Both combination techniques are applied through a 10-fold cross validation scheme on two set of classifiers,

$$\text{Set}_1 = \{C_1, \ldots, C_5\} \text{ and } \text{Set}_2 = \{C_1, \ldots, C_7\},$$

where the individual classifiers are the followings:

1. C_1 Multinomial Regression [4]: `R`'s `nnet` package;
2. C_2 Random Forest [8] : `R`'s `randomForest` package.
3. C_3 Decision Tree [9]: `R`'s `tree` package;
4. C_4 Linear Discriminant Analysis [4] `R`'s `MASS` package;

5. C_5 Shrinkage Discriminant Analysis [1]: `R`'s `sda` package;
6. C_6 kNN [4], $k = 5$: `R`'s `kknn` package;
7. C_7 SVM [29]: `R`'s `e1071` package;

All individual classifiers C_k share the same training set. Our goal is to compare HBCC with majority voting regardless of individual classifiers' accuracy and their mutual dependence. The tested data sets are described in Table 1.

4.1 Experiment Setup

All hyperparameters $\boldsymbol{\lambda}$ and $\boldsymbol{\nu}$ are set to 1. So, $\boldsymbol{\lambda}$ becomes a $J \times J$ matrix of 1 and $\boldsymbol{\nu}$ becomes a vector composed of J elements of 1. As described in 3.2, we approximate the distribution of $\mathcal{P}(\theta|\mathcal{D})$ by drawing m observations via Gibbs sampling. For each data set, we draw a sample of $m = 150$ observations, with a thinning of 20, from $\mathcal{P}(\theta|\mathcal{D})$. We noted that while applying HBCC on most data sets, the combination accuracy begun to stabilize for $m \geq 100$. This early stability of HBCC accuracy and the approximate convergence of MCMC chain motivated the choice of $m = 150$. For the sake of simplicity and due to the large sample size of some data sets, we keep m constant (150) for all data sets. We finally use Eqs. (6) and (7) to estimate the mode of true class label t^* on instances belonging to the test set.

4.2 Results and Discussion

The differences in method accuracy were verified with a McNemar test and the bold values represent cases where the method superiority was statistically significant at the 0.05 level. We can see that HBCC outperforms, majority voting on the Letter Recognition and Satellite data sets where the differences between methods accuracy are very important. On the other hand the superiority of majority voting to HBCC is significant in one combination of Vehicle data set, but this difference is not as important as in previous cases. Moreover, the application of a two-sided Wilcoxon signed rank test[3], reveals significant (`p-value=0.009`).

Table 4 reports mean and standard deviation of the classification accuracy for the seven individual classifiers (belonging to $\text{Set}_1 = \{C_1, \ldots, C_5\}$ and $\text{Set}_2 = \{C_1, \ldots, C_7\}$) tested in a 10-fold cross validation scheme. While comparing Tables 3 and 4, one could observe that the best individual classifier accuracy is usually a bit higher than both HBCC and majority voting. This superiority could be explained by the correlation between classifiers output. Indeed we suppose that the individual classifiers are conditionally independent. However, this is usually not the case, because the classifiers are build on the same variables and training sets. Furthermore, we can observe that these differences are in most cases not significant and become negligible for almost all combinations of seven classifiers[4].

[3] Wilcoxon signed rank test with continuity-correction; the alternative hypothesis was: the mean accuracy of HBCC is equal to the mean accuracy of majority voting.

[4] Look at the difference between bold values in Table 4 and HBCC or majority voting results in Table 3.

One of the main functional difference between HBCC and majority voting, is its robustness to the introduction of weak classifiers. The combinations of five classifiers $\text{Set}_1 = \{C_1, \ldots, C_5\}$ of the Letter recognition and Satellite data sets in Table 4, are good examples for such situations. In these cases the best individual classifiers are considerably more accurate than the remaining four classifiers. Actually, the robustness of HBCC to weak classifiers effect is due to its confusion matrices. The confusion matrices capture the classifiers performance and use them while combining the individual classifiers output, while in majority voting, all classifiers are treated similarly.

In Table 2, we report the average CPU times of HBCC on all data sets. Note that we measure the average time required to combine the classifiers output for one new observation which is not the same as the time required to build a combination model. The average CPU times of majority voting[5] is approximately 0.0006 for all data sets. By creating two separate orderings of data set names, one by their respective CPU times in Table 2 and the other by their respective number of classes in Table 1, we obtain the same ordering. Therefore, the reported CPU times are functions of the number of classes and they are in the same line with the computational complexity of the HBCC models due to the averaging (BMA) over all the instances of θ (here $m = 150$ instances).

The HBCC time complexity on the tested data sets are much more larger than the negligible average 0.0006 s of majority voting. However, it still remains acceptable for most of existing classification system use, if the system requires only a few prediction per second. For want of anything better, we could parallelize the computation or consider Maximum A Posteriori (MAP) predictions instead of the BMA in Eq. (6).

Table 1. Benchmark classification data sets accessible from [15].

Data set	Size	Predictor	# class
Letter recognition	20000	17	26
Satellite	6435	35	6
DNA	3186	180	3
Vehicle	846	19	4
Breast cancer	699	11	2
Iris	150	5	3

[5] The computational complexity of majority voting is $O(JK)$, where J and K are respectively the number of the true class labels and the number of individual classifiers used in the combination.

Table 2. Average CPU time of one prediction by a combination using HBCC with Set_2 on all the experimental data sets listed in Table 1, where voting has a constant average time of 0.0006. These CPU times are obtained with R version (3.1.2) on an Intel Xeon CPU E3-1220 3.10 GHz (we used a single-threaded code) with 4 GB of RAM.

Data set	HBCC CPU time
Letter recognition	0.39
Satellite	0.07
DNA	0.04
Vehicle	0.05
Breast cancer	0.03
Iris	0.04

Table 3. Comparison of majority voting (Voting) and HBCC on Set_1 and Set_2. The mean and standard deviation (shown in parenthesis) of each combination accuracy among the 10 different folds are reported. The bold cases represent a superior accuracy statistically significant with a McNemar test at the 0.05 level.

Data set	Set_1		Set_2	
	Voting	HBCC	Voting	HBCC
Letter recognition	73.8 (1.1)	**93.5** (0.8)	89 (0.8)	**96.6** (0.4)
Satellite	85.1 (1.8)	**90.7** (1.5)	88.5 (1.8)	**91.7** (1.6)
DNA	95.6 (1.3)	95.8 (0.9)	96.1 (1.5)	96 (1.1)
Vehicle	78.2 (4)	76.2 (2.9)	**78.7** (4.7)	75.6 (4.8)
Breast cancer	96.8 (2.6)	96.8 (2.4)	97.1 (2.5)	97.4 (2.4)
Iris	93.3 (4.4)	93.3 (4.4)	97.3 (3.4)	97.3 (3.4)

Table 4. Mean and standard deviation (shown in parenthesis) of each individual classifier accuracy in a 10-fold cross validation scheme. Star signs and bold values represent respectively maximum in Set_1 and Set_2.

Data set	C_1	C_2	C_3	C_4	C_5	C_6	C_7
Letter recognition	64.5 (2.7)	**96.7*** (0.4)	37.6 (1.3)	70.2 (0.9)	70.3 (0.9)	95.6 (0.3)	94.5 (0.4)
Satellite	76 (2)	**91.8*** (1.7)	80.9 (1.1)	83.8 (1.6)	83.9 (1.6)	90.3 (1.2)	89.8 (1.4)
DNA	93.4 (1.7)	95.5* (1)	90.6 (1.6)	94.6 (1.8)	94.7 (1.5)	72.5 (1.9)	**96** (1.3)
Vehicle	**78.8*** (4.7)	75.1 (4.6)	70.9 (4)	78.4 (4.1)	78.7 (5.9)	69.5 (3.4)	76.8 (5.4)
Breast cancer	96.9 (2.4)	**97.5*** (2.2)	95.3 (2.5)	96 (3.2)	96 (3.2)	96.9 (2.5)	96.5 (2.5)
Iris	94 (7.3)	94.7 (5.3)	94 (5.8)	**98*** (3.2)	98 (3.2)	93.3 (7)	96.7 (4.7)

5 Conclusion

This paper proposed a new hierarchical Bayesian method for combining the output of several classifiers. The proposed method, called Hierarchical Bayesian Classifier Combination (HBCC) is a hierarchical Bayesian model for combining the output of several discrete classifiers.

The experimental part involved the application of HBCC to six benchmark classification data sets and it shown that HBCC has generally a higher prediction accuracy than the classical majority voting. These accuracy differences were further confirmed via statistical hypothesis testing. On the other hand, the best individual classifier accuracy was usually a bit more accurate than both HBCC and majority voting. This superiority could be explained by the correlation between classifiers being in contradiction with our assumptions. Therefore, the extension of HBCC for modelling dependencies among individual classifiers seems an interesting future work. Further exploration of HBCC to classifiers with probability distribution is another avenue for research.

From an operational point of view, the HBCC time complexity depends on the data set complexity and can be considerably larger than the constant and negligible time of majority voting. However, it still remains functional for most of existing classification systems requiring no more than a few prediction recognition per second.

References

1. Ahdesmäki, M., Strimmer, K., et al.: Feature selection in omics prediction problems using cat scores and false nondiscovery rate control. Ann. Appl. Stat. **4**(1), 503–519 (2010)
2. Bauer, E., Ron, K.: An empirical comparison of voting classification algorithms: bagging, boosting, and variants. Mach. Learn. **36**(1–2), 105–139 (1999)
3. Bernardo, J.M., Smith, A.F.M.: Bayesian Theory, vol. 405. Wiley, Hoboken (2009)
4. Bishop, C.M., et al.: Pattern Recognition and Machine Learning, vol. 4. Springer, New York (2006)
5. Bolstad, W.M.: Introduction to Bayesian Statistics. Wiley, Hoboken (2013)
6. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996)
7. Breiman, L.: Stacked regressions. Mach. Learn. **24**(1), 49–64 (1996)
8. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
9. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth, Belmont (1984)
10. Clarke, B.: Comparing bayes model averaging and stacking when model approximation error cannot be ignored. J. Mach. Learn. Res. **4**, 683–712 (2003)
11. Dawid, A.P., Skene, A.M.: Maximum likelihood estimation of observer error-rates using the EM algorithm. Appl. Stat. **28**, 20–28 (1979)
12. Dietterich, T.G.: Ensemble methods in machine learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45014-9_1
13. Domingos, P.: Bayesian averaging of classifiers and the overfitting problem. In: ICML, pp. 223–230 (2000)

14. Džeroski, S., Ženko, B.: Is combining classifiers with stacking better than selecting the best one? Mach. Learn. **54**(3), 255–273 (2004)
15. Frank, A., Asuncion, A.: UCI machine learning repository (2010)
16. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. ICML **96**, 148–156 (1996)
17. Haitovsky, Y., Smith, A., Liu, Y.: Modelling disagreements among and within raters' assessments from the bayesian point of view. In: Draft. Venue: Presented at the Valencia Meeting (2002)
18. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E.: Adaptive mixtures of local experts. Neural Comput. **3**(1), 79–87 (1991)
19. Jordan, M.I., Jacobs, R.A.: Hierarchical mixtures of experts and the EM algorithm. Neural Comput. **6**(2), 181–214 (1994)
20. Kim, H.-C., Ghahramani, Z.: Bayesian classifier combination. In: International Conference on Artificial Intelligence and Statistics, pp. 619–627 (2012)
21. Lacoste, A., Marchand, M., Laviolette, F., Larochelle, H.: Agnostic Bayesian learning of ensembles. In: Proceedings of the 31st International Conference on Machine Learning, pp. 611–619 (2014)
22. Maclin, R., Opitz, D.: An empirical evaluation of bagging and boosting. AAAI/IAAI **1997**, 546–551 (1997)
23. Minka, T.P.: Bayesian model averaging is not model combination, pp. 1–2 (2000). http://www.stat.cmu.edu/minka/papers/bma.html
24. Monteith, K., Carroll, J.L., Seppi, K., Martinez, T.: Turning Bayesian model averaging into Bayesian model combination. In: The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 2657–2663. IEEE (2011)
25. Opitz, D., Maclin, R.: Popular ensemble methods: an empirical study. J. Artif. Intell. Res. **11**, 169–198 (1999)
26. Quinlan, J.R.: Bagging, boosting, and c4. 5. In: AAAI/IAAI, vol. 1, pp. 725–730 (1996)
27. Raykar, V.C., Yu, S., Zhao, L.H., Valadez, G.H., Florin, C., Bogoni, L., Moy, L.: Learning from crowds. J. Mach. Learn. Res. **11**, 1297–1322 (2010)
28. Rokach, L.: Ensemble-based classifiers. Artif. Intell. Rev. **33**(1–2), 1–39 (2010)
29. Schölkopf, B., Smola, A.: Support vector machines. In: Encyclopedia of Biostatistics (1998)
30. Simpson, E., Roberts, S., Psorakis, I., Smith, A.: Dynamic Bayesian combination of multiple imperfect classifiers. In: Guy, T., Karny, M., Wolpert, D. (eds.) Decision Making and Imperfection, vol. 474, pp. 1–35. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36406-8_1
31. Ting, K.M., Witten, I.H.: Stacking bagged and dagged models. In: ICML, pp. 367–375. Citeseer (1997)
32. Todorovski, L., Džeroski, S.: Combining classifiers with meta decision trees. Mach. Learn. **50**(3), 223–249 (2003)
33. Tulyakov, S., Jaeger, S., Govindaraju, V., Doermann, D.: Review of classifier combination methods. In: Marinai, S., Fujisawa, H. (eds.) Machine Learning in Document Analysis and Recognition, vol. 90, pp. 361–386. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-76280-5_14
34. Wolpert, D.H.: Stacked generalization. Neural Netw. **5**(2), 241–259 (1992)
35. Xu, L., Krzyzak, A., Suen, C.Y.: Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Trans. Syst. Man Cybern. **22**(3), 418–435 (1992)
36. Yuksel, S.E., Wilson, J.N., Gader, P.D.: Twenty years of mixture of experts. IEEE Trans. Neural Netw. Learn. Syst. **23**(8), 1177–1193 (2012)

Optimizing Support Vector Regression with Swarm Intelligence for Estimating the Concrete Compression Strength

Manoel Alves de Almeida Neto(✉), Roberta de Andrade de A. Fagundes(✉), and Carmelo J. A. Bastos-Filho(✉)

University of Pernambuco (UPE), Recife, Pernambuco, Brazil
maan@ecomp.poli.br, roberta.fagundes@upe.br, carmelofilho@poli.br

Abstract. Estimating the compression strength of concrete is a complex problem which has been studied by various researchers. Support Vector Regression (SVR) is a technique that has been shown to be adequate for estimation through input examples. In this paper, we assess three swarm algorithms, Fish School Search (FSS), Artificial Bee Colony (ABC) and Particle Swarm Optimization (PSO) aiming to optimize the SVR parameter. The results show that both all swarm-based algorithms far outperformed the original SVR in the concrete compression strength estimation task and the FSS and ABC obtained better results than PSO for this particular problem.

Keywords: Regression models · Concrete compression strength
Hybrid algorithms · Swarm intelligence

1 Introduction

For estimating values ($\widehat{y}$) from input data, one of the well-known and widely used techniques is regression analysis. Regression analysis is a statistical tool which aims to explain the relationship between variables through mathematical models. The variables are divided into two groups: independent (x); and dependent variables (y) [1]. There are various kinds of regression analysis which are applied to different types of problem. They are classified in parametric (i.e., linear) and nonparametric (i.e., nonlinear) regression models. For parametric models, the well known technique is Linear regression [2], while for the nonparametric, we can cite Support Vector Regression Kernel Regression (SVR) [3]. The biggest difference between parametric and nonparametric techniques is the use of a kernel function.

Swarm Intelligence is an area of Computational Intelligence that encompasses several bio-inspired algorithms developed for solving optimization problems. These algorithms are based on socio-biological behavior to search food and run away from predators. Particle Swarm Optimization (PSO) developed by Kennedy [4] is one of the well-known algorithms in swarm intelligence. PSO has a significant exploitation capability. On the other hand, Artificial Bee Colony (ABC) developed by Karaboga [5] is another algorithm widely deployed for solving optimization problems. ABC has an interest ability to solve problems in multi-modal spaces. Besides, other swarm-based algorithms have been developed aiming to present the capability to self-regulate exploitation and exploration, e.g., Fish School Search (FSS). FSS was inspired by the fish schools searching for food [6].

Estimating the compression strength of the high-performance concrete (HPC) is a complex problem to solve in civil engineering because there are some essential ingredients such as water, fine and coarse aggregates whose the concentration affects directly in the quality of the HPC. Also, the process to make HPC needs other supplementary materials, such as fly ash and blast furnace slag, and chemical admixture, such as super-plasticize [7]. Since 1918, the most accepted way to calculate the strengths compression of concrete is the Abrams' water-to-cement ratio (w/c). It's a formulation in which the increasing of w/c implies in decreasing of the concrete strength and vice-versa. However, a lot of studies have shown that HPC strength does not depend only on the w/c ratio, but it is also influenced by other factors [8,9].

This paper aims to assess four different ways to build regression models for predicting the compression strength of HPC using support vector regression (SVR). The four proposal are optimized by FSS, ABC, and PSO (Local and Global), respectively.

The remainder of the paper is divided as follows. Section 2 describes the background. Section 3 presents the proposed models to predict the compression strength of concrete. Sections 4 and 5 present the experimental setup and the obtained results, respectively. Finally, Sect. 6 shows the conclusions and future works.

2 Background

Regression models have been widely used for predicting values in several areas, for instance, software defect estimation using support vector regression [10] and applied in the microcystin concentrations in lakes and reservoirs at a continental scale [11]. However, a predicting model without the proper adjustment is not a compelling technique since most of them have parameters that can cause a drastic impact on the performance of the model without a correct calibration. Besides, it is tough to find the best values for the parameters for each technique while avoiding overfitting or underfitting since the number of combinations of the values for the parameters can be infinity. As an example, we can cite the SVR which presents the parameters ϵ, C and γ (Kernel function parameters) that are continuous variables.

Swarm-based optimization algorithms have been proposed and used widely focusing on solving optimization problems with many continuous variables. Particle Swarm Optimization (PSO) is an algorithm inspired by flocks of birds searching for foods [4]. Among many examples of this type of use of the PSO, we can cite a hybrid particle swarm optimization and genetic algorithm for closed-loop supply chain network design in large-scale networks [12,13] which used analytical selection to optimize SVR parameters to get the better accuracy based on a priori approach about these parameters.

Fish School Search (FSS) algorithm was inspired by the collective behavior of fish schools [6]. FSS is suitable for optimization problems with multi-modal search spaces. Among the examples of applications of FSS in hybrid models, we can cite the combination of K-Means and K-Harmonic with FSS for data clustering task on graphics processing units [14].

Artificial Bee Colony (ABC) algorithm was developed by mimicking of the organization of the bee colonies [5]. Such as FSS, ABC is good at solving problems with various minimal and maximal solutions. For instance, ABC was hybridized to symbolic regression to find the best combination of variables, symbols, and coefficients that resulted in suitable solutions [15].

2.1 Support Vector Regression

Support Vector Regression (SVR) was developed by Drucker et al. for solving regression problems [3] with the same proposal of the Support Vector Machine (SVM), generating the maximum number of support of vectors with small values of errors aiming to separate the data with the highest margin. SVR parameters are: ϵ, which indicates the level of tolerance that the support vectors can have regarding real data; and C, which is the cost function to generate the support vectors. To evaluate nonparametric regression operations, SVR uses a Kernel function. A kernel is a mathematical approach for estimating density curves in which each observation is weighted by the distance from a central value [16]. The bandwidth, defined by the variable γ, is a free Kernel parameter that represents a substantial influence on the estimating result of the regression model. Equation (1) shows how the SVR works, whereas Eq. (2) is a simplification of Kernel Radial Basis Function.

$$\min \frac{1}{2}||w||^2 + C\sum_{n=1}^{n}(\epsilon_i + \epsilon_i^*), \begin{cases} y_i - wx_i - b \leq \epsilon + \epsilon_i \\ wx_i + b - y \leq \epsilon \end{cases} \tag{1}$$

$$K(x, x^*) = \begin{cases} \exp\left(-\frac{||x-x^*||}{2\sigma}^2\right), & \gamma = \frac{1}{2\sigma}^2 \\ \exp(-\gamma||x - x^*||^2) \end{cases} \tag{2}$$

Support Vector Regression presents advantages over other techniques since it focuses on finding the global optimal and the resultant model is easier to understand than other famous approaches, such as Artificial Neural Network.

2.2 Fish School Search

Fish School Search (FSS) algorithm was inspired to mimic the behavior of fish schools in searching for food and getting away from predators [6]. The FSS model is composed of three types of movements: individual; collective-instinctive; collective-volitive.

In the individual movement, every fish in the school tries to find locally and greedy a better solution in the neighborhood. After that, it is executed the feeding process individually. Then, it is calculated a weighted average of those fishes which became heavier (fittest) than other ones for collective-instinctive movement.

In the collective-instinctive movement, each fish is repositioned following the preferential directions influenced by the fish that had better performed in individual movement.

Finally, after individual and collective-instinctive movement have been performed, we execute the collective-volitive movement. In this stage, we evaluate the overall performance of the fish school. Thus, we calculate the weight of the fish school for verifying whether the weight has increased after the execution of the three movements in the last iteration. If the school became heavier, then the fish school will contract, i.e., it intensifies the exploitation capability. Otherwise, it expands the radius of the swarm, thus increasing the exploration behavior. This last movement gives an advantage over other optimization algorithms in the search processes since it self-regulates the balance between exploration or exploitation during the search process.

2.3 Artificial Bee Colony

Artificial Been Colony (ABC) algorithm was inspired by the collective behavior that the bees do in searching for source foods, collecting and deciding when any source food has to be abandoned [5]. *Waggle Dance* is a communicating mechanism in which the bees use to explain where there are food sources and how distant it is from the niche.

In a nest, the bees are organized in three different groups: employed, onlookers and scouts. Employed bees use *Waggle Dance* to guide the onlooker bees telling them the coordinates (direction and distance) and the quality of each source food. Thus, the onlooker bees decide whether it is worth to explore the food source or not. In the case of choosing one the food source, the onlookers will keep extracting food until there is not more food or it is not worth to keep extracting food. Otherwise, the onlookers will be waiting until a new food source arises. Scouts bees are responsible for searching new food source randomly. The number of food sources (employed or onlooker bees) in ABC algorithm is equal to the number of solutions in the swarm.

2.4 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is an algorithm which mimics the behavior of the flock of birds [4]. It is a population-based algorithm with simple reactive agents (particles), where the position of each particle represents a possible solution for an optimization problem. Each particle has four attributes. A vector describing the position in the search-space; a velocity vector, responsible for guiding and controlling the movement of the particles; the best position that the particles found along the search process; and the best position obtained from the neighbor particles.

Focusing on improving PSO algorithm, studies have been proposing new communication typologies to overcome some limitations. A communication topology defines how the particles could exchange information with each other, i.e., it determines the neighborhood of the particles. The well-known topology is the global. In the Global topology, particles of the swarm are allowed to communicate to all the other particles. Thus, the whole flock can share the same best position *gbest* found so far by the best particle. The distribution of the global knowledge leads to a fast convergence, but in some cases can lead to a premature convergence.

Local topologies, such as the ring, have been developed to avoid these stagnation processes. In this topology, the population is not fully connected, and there appear various sub-groups, each one running as a separate swarm with its own local neighborhood best. Inside each neighborhood, there is the best particle called *lbest*. Since the particles are loosely connected, the speed of convergence is significantly reduced. Thus, it avoids the algorithm to get trapped in local minima easily.

Finally, the movement of each particle is defined by a velocity function, for instance, inertia and Clerc factor construction [17].

3 Proposal of the Hybrid Algorithms

In this paper, we assess three swarm-based algorithms to optimize the SVR parameters, ϵ and C, and Kernel parameter γ, which controls the bandwidth of the Radial Basis Function (RBF). For each case, we have a hybrid model, named FSS-SVR, ABC-SVR, PSO/Local-SVR and PSO/Global-SVR. We deployed the Root Mean Squared Error (RMSE) as the algorithm evaluation metrics since it represents the standard deviation value of the residuals of the regression (R^2). In additional, RMSE measures how separate the residuals are distributed from the real data and its value is always a number between 0 and 1, 0% to 100% [18].

PSO has an excellent ability for performing exploitation in the search process, achieving high precision in search-spaces with few local optima. Thus, we expect to obtain a higher precision for the values of the SVR parameters when performing the hybridization of this algorithm. We chose to use the global and local communication topologies. For the velocity function, we used the constriction factor [17].

The ABC algorithm can present high performance when solving problems in multi-modal spaces because it has mechanisms to divide the swarm into two groups. One is responsible for exploring promising spots in the search space, and then the second aims to extract as much information as possible (i.e., exploitation) [15].

Finally, in the third attempt, we used the FSS because it has the capability of automatically change from exploitation to exploration and vice-versa. Thus, it could get off from minimal and maximal local more efficiently. For the ABC and FSS, we expect to have a better exploration of the values of the SVR parameters in the search-space. Algorithm 1 shows how the hybridized algorithms work.

Algorithm 1. Pseudo-code of the hybridized algorithms - PSO-SVR/ABC-SVR/FSS-SVR.

```
0: Initialize the swarm algorithm parameters.
1: while stopping condition is not achieved do
1:     Set $\epsilon$, $C$ (Equation (1)) and $\gamma$ (Equation (2)) to SVR for predicting the Concrete
       Compress Strength.
1:     Generate the Support Vectors according to Equation (1).
1:     Calculate the Root Mean Squared Error (fitness).
2: end while
2: Return optimal solution.
```

4 Experimental Setup

We used a real-world data set to compare the hybrid proposed algorithms (FSS-SVR, ABC-SVR, PSO/Local-SVR and PSO/Global-SVR), called Concrete Compressive Strength. This is a nonparametric data set, and the goal is to predict the concrete compressive strength [7].

The dictionary of the data set is described in Table 1. We used the data normalization approach to set all the data in the same scale, between 0.15 and 0.85.

Table 1. Concrete Compressive Strength (CCS) - data set dictionary

Data type	Name	Measurement	Descriptions
Double	Cement	KG for m^3	Input
Double	Blast Furnace Slag	KG for m^3	Input
Double	Fly Ash	KG for m^3	Input
Double	Water	KG for m^3	Input
Double	Superplasticizer	KG for m^3	Input
Double	Coarse Aggregate	KG for m^3	Input
Double	Fine Aggregate	KG for m^3	Input
Double	Age	Day (1 365)	Input
Double	CCS	MPa	Output

We used the following values for the SVR parameters and for the FSS-SVR, ABC-SVR, PSO/Local-SVR and PSO/Global-SVR approaches:

- SVR canonical: $\epsilon = 1.0\text{E}{-12}$, $C = 1$ and $\gamma = 0.01$.
- PSO/Global-SVR, PSO/Local-SVR, ABC-SVR and FSS-SVR: ϵ [1.0E−13, 10.0], C [1.0E−10, 50] and γ [0.0001, 1].

We executed all the experiments as follows. We performed 30 independent executions for each algorithm, SVR, FSS-SVR, ABC-SVR, PSO/Local-SVR and PSO/Global-SVR. We used 30 particles for the PSO/Global-SVR and PSO/Local-SVR. We deployed 15 onlooker bees and 15 employed bees for ABC-SVR. We used 30 fish for the FSS-SVR. We run 100 iterations considering the three dimensions for all the algorithms. For PSO/Global-SVR and PSO/Local-SVR, we used: the constriction factor equal to 0.72984; C_1 and C_2 equal to 2.05. For the ABC-SVR we used 30 tries for each food source; 40 the maximum limit for each employed bee to keep searching for food in each food sources. For the FSS-SVR, we used the following values for the parameters: $Weight_{initial}$ [300,600]; $Weight_{final}$ [2000,5000]; $Step_{indi}$ and $Step_{vol}$ [1.0,0.00001]. The dataset was divided into two parts, 70%, and 30%. 70% of the samples were used for training process, and 30% were used for testing and validating the proposed algorithms. In each iteration, all the proposed algorithms do the training, testing and validating processes. Besides, SVR focuses on finding the optimal regression for a given a combination of parameters.

5 Results

We organized the obtained results by experiments in visual graphics, convergence, box-plot and predicted versus real data charts. We also provide one table containing the descriptive statistical values. Besides, since all the results obtained by the algorithms after 30 executions did not follow a normal distribution, we applied the *Wilcoxon* test for comparing which hybrid technique performed better.

At first, we evaluated the canonical version of the SVR and we obtained Root Mean Squared Error (RMSE) equal to 0.1014. It is worthy to highlight that there is no standard deviation since the SVR is deterministic for a predefined data set. However, since we perform a stochastic optimization process in the FSS-SVR, ABC-SVR, PSO/Local-SVR and PSO/Global-SVR cases, the RMSE can vary for each simulation. We obtained an average RMSE of 0.02483, 0.02469, 0.02488 and 0.02486 for FSS-SVR, ABC-SVR, PSO/Local-SVR and PSO/Global-SVR, respectively. The Fig. 1 shows the box-plot of the RMSE and Table 2 presents all the statistical indicators, such as the minimal, maximal, median, mean and standard deviation for the RMSE. Since the canonic SVR is deterministic, all the indicators assume 0.1014 and the standard deviation is 0. Besides, given that the canonical SVR achieved a lousy result with no variance, we only consider the other assessed algorithms in the further analysis.

We also analyzed the convergence of the optimization algorithms. The results presented in Fig. 2 are for the minimal error obtained for FSS-SVR = 0.02458, ABC-SVR = 0.02396, PSO/Local-SVR = 0.02478 and PSO/Global-SVR = 0.02476. One can observe that, as it was expected, PSO/Global-SVR converged

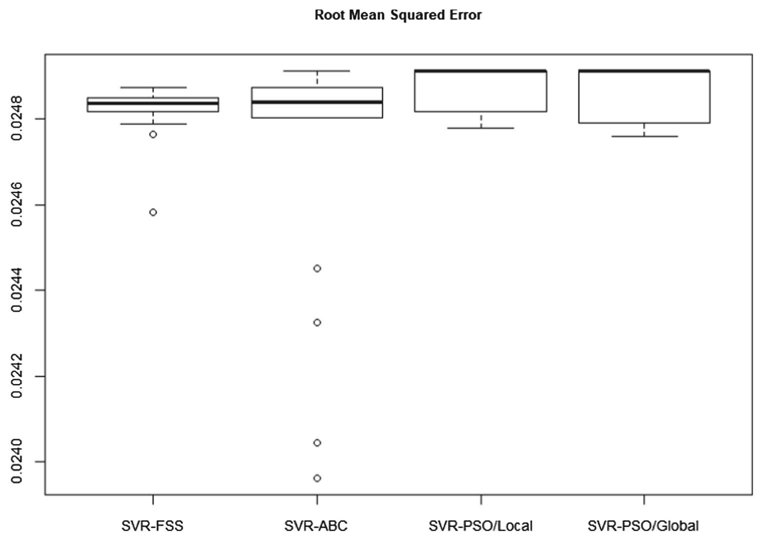


Fig. 1. Box-plot of the errors of the FSS-SVR, ABC-SVR, PSO/Local-SVR and PSO/Global-SVR

faster than the other ones and got stuck after five iterations. Differently for the PSO/Global-SVR, the rest of the hybridized algorithms were able to find better solutions in the search space. ABC-SVR converged after 28 iterations, PSO/Local-SVR at 39 iterations and FSS-SVR after 58 iterations. As previously mentioned, PSO Local and Global, ABC and FSS have different behaviors in the search process. PSO/Global converges faster and focuses more on the exploitation of promising areas of the search space. However, even ABC is good at multi-modal problems, it had a similar behavior to PSO with global topology converging to an optimal point fast, but it found better results than PSO. PSO/Local and FSS were the slowest in converging because they spent more time exploring the search space aiming to find promising areas. In this last case, we highlight the FSS since it explored more the search space, thus resulting in a better performance than PSO/Local.

Table 2. SVR, FSS-SVR, ABC-SVR, PSO/Local-SVR, PSO/Global-SVR - statistical results

Metric	SVR	FSS-SVR	ABC-SVR	PSO/Local-SVR	PSO/Global-SVR
Minimal	0.1014	0.02458	0.02396	0.02478	0.02476
Maximal	0.1014	0.02487	0.02491	0.02491	0.02491
Median	0.1014	0.02484	0.02484	0.02491	0.02491
Mean	0.1014	0.02483	0.02469	0.02488	0.02487
Stand. Dev	0	5.14E−05	0.000322932	4.93E−05	6.34E−05

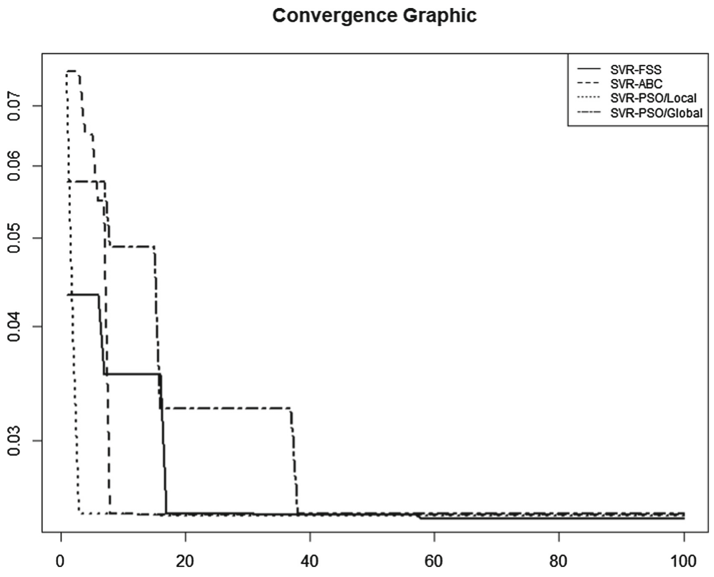
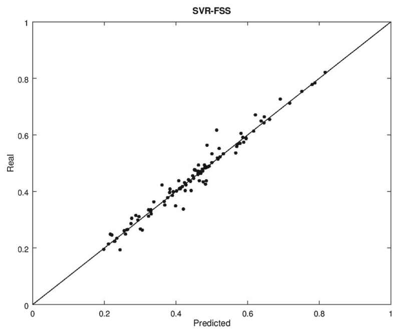


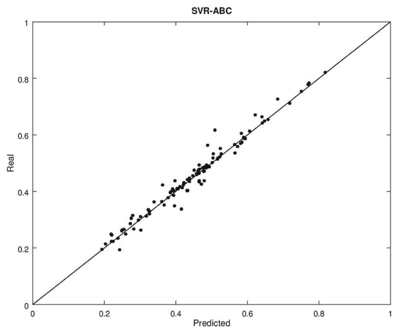
Fig. 2. Convergence graphic of the proposed algorithms

We analyze the real data versus estimated data generated by FSS-SVR, ABC-SVR, PSO/Local-SVR and PSO/Global-SVR. The results presented in Fig. 3 were obtained from the same executions used to compose Fig. 2. It is worthy to mention that a perfect regression occurs when it is possible to draw a straight perpendicular between the data. As it can be noted, Fig. 3(a) and (b) from FSS-SVR and ABC-SVR have more points near to the straight, which means that both of them overcame the hybridized algorithms based on PSO, i.e., PSO/Local-SVR and PSO/Global-SVR, Fig. 3(c) and (d), respectively.

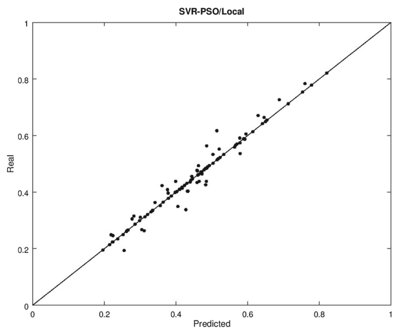
Finally, we present the result obtained for the nonparametric *Wilcoxon* hypothesis test for the RMSE. Since *Kolmogorov-Smirnov* test indicated that the results do not fit a Gaussian distribution [19], we used the signed ranked *Wilcoxon* nonparametric test. We tested the results following this order: FSS-SVR x ABC-SVR; FSS-SVR x PSO/Local-SVR; FSS-SVR x PSO/Global-SVR; ABC-SVR x PSO/Local-SVR; ABC-SVR x PSO/Global-SVR. For the null hypothesis (h_0), we verify if the algorithms are identical, and for alternative hypothesis (h_1), if one of the algorithms has a lower median value than the other one. Therefore, we obtained the p-value equal to 0.761796252, 0.00017591, 0.007492762, 0.000103564 and 0.001319936, for FSS-SVR x ABC-SVR; FSS-SVR x PSO/Local-SVR; FSS-SVR x PSO/Global-SVR; ABC-SVR x PSO/Local-SVR; ABC-SVR x PSO/Global-SVR, respectively. Thus, we can conclude that all the hybridized techniques overperformed the canonical SVR. Besides, there is no statistical difference between FSS-SVR and ABC-SVR, and they were better than PSO/Local-SVR and PSO/Global-SVR for considering a 95% of confidence level.



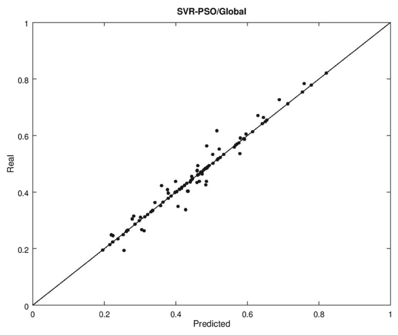
(a) Results of FSS-SVR.



(b) Results of ABC-SVR.



(c) Results of PSO/Local-SVR.



(d) Results of PSO/Global-SVR.

Fig. 3. Comparison between all the results obtained by the SVR, PSO-SVR and FSS-SVR

6 Conclusion

We demonstrated that it is possible to use swarm-based algorithms, such as FSS, ABC, and PSO, in a hybridization process to accurately find the suitable values for the SVR parameters. The results showed an excellent accuracy gain for estimating concrete compression strength (CCS). The root mean squared error obtained after the hybridization is smaller than the original SVR, and the FSS-SVR and ABC-SVR achieved the best results. For future works, we intend to analyze how the SVR parameters influence the outcomes of the predictions and plan to assess the utilization of another Kernel functions, such as polynomial and tangent sigmoid kernels.

References

1. Montgomery, D.C.: Introduction to Statistical Quality Control. Wiley, Hoboken (2007)
2. Huang, H.H., Huang, S.Y.: Nonlinear regression analysis. In: International Encyclopedia of Education, pp. 339–346 (2010)
3. Basak, D., Pal, S., Patranabis, D.C.: Support vector regression. Neural Inf. Process.-Lett. Rev. **11**(10), 203–224 (2007)
4. Kennedy, J.: Particle swarm optimization. In: Gass, S.I., Fu, M.C. (eds.) Encyclopedia of Machine Learning, pp. 760–766. Springer, Boston (2013). https://doi.org/10.1007/978-1-4419-1153-7_200581
5. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)
6. Bastos Filho, C.J., de Lima Neto, F.B., Lins, A.J., Nascimento, A.I., Lima, M.P.: Fish school search. In: Chiong, R. (ed.) Nature-Inspired Algorithms for Optimisation, vol. 193, pp. 261–277. Springer, Berlin (2009). https://doi.org/10.1007/978-3-642-00267-0_9
7. Yeh, I.C.: Modeling of strength of high-performance concrete using artificial neural networks. Cem. Concr. Res. **28**(12), 1797–1808 (1998)
8. Almeida Filho, F., Barragán, B.E., Casas, J., El Debs, A.: Hardened properties of self-compacting concrete—a statistical approach. Constr. Build. Mater. **24**(9), 1608–1615 (2010)
9. Aslani, F., Nejadi, S.: Mechanical properties of conventional and self-compacting concrete: an analytical study. Constr. Build. Mater. **36**, 330–347 (2012)
10. de A. Fagundes, R.A., de Souza, R.M.C.R.: Software defect estimation using support vector regression. In: Proceedings of the 22nd International Conference on Software Engineering & Knowledge Engineering (SEKE 2010), Redwood City, San Francisco Bay, CA, USA, 1–3 July 2010, pp. 265–268 (2010)
11. Taranu, Z.E., Gregory-Eaves, I., Steele, R.J., Beaulieu, M., Legendre, P.: Predicting microcystin concentrations in lakes and reservoirs at a continental scale: a new framework for modelling an important health risk factor. Glob. Ecol. Biogeogr. **26**(6), 625–637 (2017)
12. Soleimani, H., Kannan, G.: A hybrid particle swarm optimization and genetic algorithm for closed-loop supply chain network design in large-scale networks. Appl. Math. Model. **39**(14), 3990–4012 (2015)
13. Zhao, W., Tao, T., Zio, E., Wang, W.: A novel hybrid method of parameters tuning in support vector regression for reliability prediction: particle swarm optimization combined with analytical selection. IEEE Trans. Reliab. **65**(3), 1393–1405 (2016)
14. Serapião, A.B., Corrêa, G.S., Goncalves, F.B., Carvalho, V.O.: Combining k-means and k-harmonic with fish school search algorithm for data clustering task on graphics processing units. Appl. Soft Comput. **41**, 290–304 (2016)
15. Karaboga, D., Ozturk, C., Karaboga, N., Gorkemli, B.: Artificial bee colony programming for symbolic regression. Inf. Sci. **209**, 1–15 (2012)
16. Rosenblatt, M.: Remarks on some nonparametric estimates of a density function. In: The Annals of Mathematical Statistics, pp. 832–837 (1956)
17. Clerc, M., Kennedy, J.: The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6**(1), 58–73 (2002)

18. Barnston, A.G.: Correspondence among the correlation, RMSE, and Heidke forecast verification measures; refinement of the Heidke score. Weather Forecast. **7**(4), 699–709 (1992)
19. Wilcoxon, F.: Individual comparisons by ranking methods. Biom. Bull. **1**(6), 80–83 (1945)

Automated Contradiction Detection in Biomedical Literature

Noha S. Tawfik[1,2(✉)] and Marco R. Spruit[2]

[1] Computer Engineering Department, College of Engineering, Arab Academy for Science, Technology, and Maritime Transport (AAST), Abukir, Alexandria 1029, Egypt
noha.abdelsalam@aast.edu

[2] Department of Information and Computing Sciences, Utrecht University, Princetonplein 5, 3584 CC Utrecht, The Netherlands
{n.s.tawfik,m.r.spruit}@uu.nl

Abstract. Medical literature suffers from inconsistencies between reported findings that answer the same research question. This paper introduces an automated two-phase contradiction detection model that integrates semantic properties as input features to a Learning-to-Rank framework, to accurately identify key findings of a research article. It also relies on negation, antonyms and similarity measures to detect contradictions between findings. The proposed technique is implemented and tested on a publicly available contradiction corpus 259 manually annotated abstracts. The performance is compared based on recall, precision and F-measure. Experimental evaluations prove the utility of the model and its contribution to the contradiction classification and extraction task.

Keywords: Biomedical NLP · Answer selection
Contradiction detection · Information extraction · Text mining

1 Introduction

In the last decade, there was a substantial increase in the total number of medical research publications worldwide. Most of the literature publish results on the effectiveness of clinical interventions, and despite the similarity of the scientific experiment designs, not all outcomes are in agreement [12]. Whether published findings are consensual, complimentary, or contradictory facts, many of them get approved, updated or replaced accordingly [23]. Given the varying nature of published findings, it is difficult to fairly assess evidence-based knowledge within articles. More importantly, differences between research outcomes should be highlighted so that further studies do not build assumptions and/or conclusions on prior research that have since been disapproved, and are not valid anymore.

In extreme cases, some published evidence-based facts even get reversed. Prasad et al. [24], reviewed 363 articles published in one high impact factor

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 138–148, 2018.
https://doi.org/10.1007/978-3-319-96136-1_12

journal investigating various established medical practices. While 138 (38%) confirmed the practices, 146 (40.2%) found them ineffective and 79 (27.3%) were inconclusive. For example, four studies contradicted the administration of the Aprotinin drug, widely used for treatment in post cardiac surgeries.

Such misinformation, or disinformation, create controversy that is important to researchers and practitioners interested in finding evidence-based answers to clinical queries; whether it is for the benefit of their patients or for the sake of conducting systematic reviews. It is also of great significance to both Comparative Effectiveness Research (CER) and the Precision Medicine (PM) communities. The comparative effectiveness research is interested in the analysis of medical interventions by comparing their benefits and drawbacks, to reach informed evidence-based decisions for a better clinical practice [29]. While Precision Medicine also aims at improving the health care system, PM is different than CRM as it takes into account the genetic, environmental and lifestyle differences between individuals [14]. Highlighting different outcomes to the same medical practice supports the PM claims that there is no "one-size-fits-all" treatment strategy.

However, with the high rate of growth in scientific publications, the task of finding answers, interpreting outcomes and validating them becomes tedious, exhausting and time consuming, even in a specific sub-domain. In result, several text mining tools and frameworks were built and employed to solve the information extraction problem, automatically or semi-automatically, for a variety of research applications. Biomedical text mining faces a number of challenges; the enormous number of existing publications, the unstructured nature of text, and most challengingly, the ambiguity of reporting biomedical or clinical results. Findings can be expressed in long, context-dependent sentences with the usage of a wide variety of terminology.

Contradiction Detection in text is still a relatively new area of research. As in other Natural Language Processing (NLP) sub-domains, it requires a multidisciplinary approach involving text mining, sentiment analysis, opinion mining, knowledge retrieval and information extraction. This paper focuses on the problem of extracting contradicting findings in biomedical texts. In this context, we propose an automated contradiction detection framework that adapts and extends existing NLP tools. The proposed model takes advantage of a recently published corpus, constructed for the same purpose, to validate its accuracy.

2 Related Work

Despite the fact that more research has been conducted on text entailment rather than contradiction detection, the development of two contradiction corporas encouraged more research into the domain [9,20,26]. The corporas were based on direct negation and paraphrasing of sentences from the PASCAL Recognizing Textual Entailment (RTE) dataset [11]. However, contradiction analysis remains a challenging task, mainly due to the different ways in which contradictions can appear (numeric mismatching, negations, contrastive sentences, etc.).

The importance of extracting contradictions has been exploited in other domains, most commonly in news and rumors text processing. Due to the generic type of negation found in normal text, it is difficult to adapt any of the former models to the biomedical domain directly. The language used to express biomedical facts is usually rich in clinical semantics and conceptual overlaps, and involves complex sentence structures. To the best of our knowledge, there has been minimal research conducted on the biomedical contradiction analysis.

In 2011, Sarafraz et al. [27] investigated both rule based and machine learning methods to identify negated molecular events through lexical, syntactic and semantic features, the model was evaluated on the BioNLP09 challenge corpus. Alamri et al. [2,3] explored the use of four features: negation, directionality sentiment and uni+bi-grams combined with an SVM classifier, to extract contrasted findings reported in cardiovascular research literature. More recently, Preum et al. [25] presented *Preclude*, a rule-based system that highlights conflicts in wellness advice, found in on-line health forums. The system constructs a polarity lexicon from verbs , in the training set, and their synonyms using WordNet, for labeling actions found in text as positive or negative. In another attempt to discover the ambiguities in the biomedical literature, de Silva et al. [28] proposed an ontology-based system to extract inconsistencies found in miRNA research articles in the PubMed repository. The system relies on OLLIE "Open Language Learning for Information Extraction" framework to extract all relevant triples (subject, object, and relationship) from abstracts. Triple entries are then compared against each other to find inconsistencies, based on an *oppositeness metric* suggested by the authors.

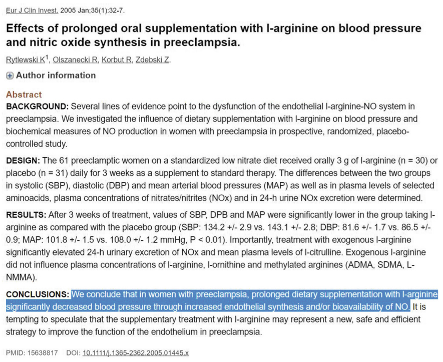
3 Dataset

The lack of annotated data has led to the unavailability of comparison and evaluation of contradiction detection systems in the biomedical literature. However, this may change with the recent availability of Manual Contradiction Corpus (*ManConCorpus*), a corpora of contradictory research claims[1]. The corpus is constructed out of 24 systematic reviews on four important cardiovascular disease topics: Cardiomyopathy, Coronary artery, Hypertensive and Heart failure. Each review article is mapped to a closed PICO (Population, Intervention, Comparison and Outcome) question that could be answered only by Yes or No. The mapping process was conducted manually by a medical expert, after reviewing all research abstracts of studies included in the systematic review. These abstracts include research claims with answers to the questions. *A research claim* is a one-sentence summary of the research findings that the authors find important, either to affirm old information or to introduce new ones. Two annotators were asked separately to find one correct claim per abstract and label it *YES*, if it positively answers the question and *NO* otherwise. It is worth mentioning that despite the fact that multiple sentences in the abstract might hold the answer

[1] Corpus available at http://staffwww.dcs.shef.ac.uk/people/M.Stevenson/resources/bio_contradictions/.

to the query, only the most informative one is chosen as per the annotator's opinion. The corpus has a total of 259 abstracts, out of which 180 introduce positive claims and 79 introduce negative claims. All claims included in the corpus are either evaluative or causal. The former is an assessment of the biomedical concept presented in the research topped by a judgment, while the latter is a statement that describes the relation type between two concepts and whether one affects the other or not. More details on the annotation process and the corpus statistics can be found in [4].

In literature, there is no standard definition of 'contradiction', and it is usually task-dependent according to the nature of the contradiction instances. Therefore, we adopt the authors' definition of contradiction that better matches the corpus and human intuitions: "*Two texts, T_1 and T_2, are said to contradict when, for a given fact F, information inferred about F from T_1 is unlikely to be true at the same time as information about F inferred from T_2*". As per the definition, if both a positive and a negative claim answer the same query, they are considered contradictory as shown in the example in Fig. 1.



Eur J Clin Invest. 2005 Jan;35(1):32-7.

Effects of prolonged oral supplementation with l-arginine on blood pressure and nitric oxide synthesis in preeclampsia.

Rytlewski K[1], Olszanecki R, Korbut R, Zdebski Z.

Author information

Abstract

BACKGROUND: Several lines of evidence point to the dysfunction of the endothelial l-arginine-NO system in preeclampsia. We investigated the influence of dietary supplementation with l-arginine on blood pressure and biochemical measures of NO production in women with preeclampsia in prospective, randomized, placebo-controlled study.

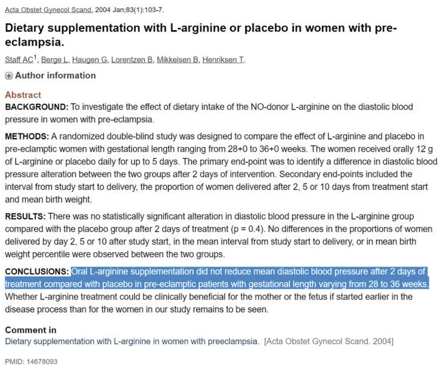
DESIGN: The 61 preeclamptic women on a standardized low nitrate diet received orally 3 g of l-arginine (n = 30) or placebo (n = 31) daily for 3 weeks as a supplement to standard therapy. The differences between the two groups in systolic (SBP), diastolic (DBP) and mean arterial blood pressures (MAP) as well as in plasma levels of selected aminoacids, plasma concentrations of nitrates/nitrites (NOx) and in 24-h urine NOx excretion were determined.

RESULTS: After 3 weeks of treatment, values of SBP, DPB and MAP were significantly lower in the group taking l-arginine as compared with the placebo group (SBP: 134.2 +/- 2.9 vs. 143.1 +/- 2.8; DBP: 81.6 +/- 1.7 vs. 86.5 +/- 0.9; MAP: 101.8 +/- 1.5 vs. 108.0 +/- 1.2 mmHg, P < 0.01). Importantly, treatment with exogenous l-arginine significantly elevated 24-h urinary excretion of NOx and mean plasma levels of l-citrulline. Exogenous l-arginine did not influence plasma concentrations of l-arginine, l-ornithine and methylated arginines (ADMA, SDMA, L-NMMA).

CONCLUSIONS: We conclude that in women with preeclampsia, prolonged dietary supplementation with l-arginine significantly decreased blood pressure through increased endothelial synthesis and/or bioavailability of NO. It is tempting to speculate that the supplementary treatment with l-arginine may represent a new, safe and efficient strategy to improve the function of the endothelium in preeclampsia.

PMID: 15638817 DOI: 10.1111/j.1365-2362.2005.01445.x

(a) PMID: 15638817 with assertion value YES



Acta Obstet Gynecol Scand. 2004 Jan;83(1):103-7.

Dietary supplementation with L-arginine or placebo in women with pre-eclampsia.

Staff AC[1], Berge L, Haugen G, Lorentzen B, Mikkelsen B, Henriksen T.

Author information

Abstract

BACKGROUND: To investigate the effect of dietary intake of the NO-donor L-arginine on the diastolic blood pressure in women with pre-eclampsia.

METHODS: A randomized double-blind study was designed to compare the effect of L-arginine and placebo in pre-eclamptic women with gestational length ranging from 28+0 to 36+0 weeks. The women received orally 12 g of L-arginine or placebo daily for up to 5 days. The primary end-point was to identify a difference in diastolic blood pressure alteration between the two groups after 2 days of intervention. Secondary end-points included the interval from study start to delivery, the proportion of women delivered after 2, 5 or 10 days from treatment start and mean birth weight.

RESULTS: There was no statistically significant alteration in diastolic blood pressure in the L-arginine group compared with the placebo group after 2 days of treatment (p = 0.4). No differences in the proportions of women delivered by day 2, 5 or 10 after study start, in the mean interval from study start to delivery, or in mean birth weight percentile were observed between the two groups.

CONCLUSIONS: Oral L-arginine supplementation did not reduce mean diastolic blood pressure after 2 days of treatment compared with placebo in pre-eclamptic patients with gestational length varying from 28 to 36 weeks. Whether L-arginine treatment could be clinically beneficial for the mother or the fetus if started earlier in the disease process than for the women in our study remains to be seen.

Comment in
Dietary supplementation with L-arginine in women with preeclampsia. [Acta Obstet Gynecol Scand. 2004]

PMID: 14678093

(b) PMID: 14678093 with assertion value NO

Fig. 1. An example of two contradictory claims found in literature that answer the query: *In women with pre-eclampsia, does treatment with L-Arginine, compared to a placebo, reduce blood pressure?*

4 Methods

Identifying inconsistencies in text is a two-phase problem, claim retrieval and claim assertion. During the first phase, we need to identify potential sentences relevant to the query. In the claim assertion phase, we have to evaluate whether sentences infer text entailment or contradiction.

4.1 Identification of Abstract Claims

Finding relevant sentences that answer the query is a key component in the biomedical contradiction detection system, as the performance of the system is dependent on the accuracy of the extracted key phrase. Several methods and techniques have been employed for passage retrieval in general, and answer identification in specific. Nevertheless, it still remains a challenge in the biomedical language processing field, mainly due to the complex nature of the text. In this research, we address the claim extraction process as a ranking problem, where each sentence in the input text is scored according to its relevance to the query.

Input Preprocessing. We split all abstract text included in the corpus into sentences using The Natural Language Toolkit (NLTK). All sentences with less than three words are considered an error of the splitting process, and thus eliminated. Afterwards, a set of potential claims that answer the query correctly is compiled for each abstract. As in any text mining application, the input text might be totally unstructured or semi-structured, and the same applies for literature abstracts. For slightly structured abstracts, i.e abstracts where text is divided into subsections such as Title, Introduction/Background, Methods/Aims, Results, and Conclusion, we take advantage of this information and include all sentences within the headings, Results and Conclusion, as candidate sentences. If the text is unstructured, all sentences in the second half of the abstract are included the candidate set, following the assumption that important findings are most probably reported by the end of the abstract. The candidate set is filtered out from any stop words, symbols and punctuations. All 24 PICO questions went through the same filtering process as the candidate sentence collection.

Feature Extraction. A fixed length feature vector representing sentences included in the candidate set is derived. These features combine both semantic and syntactic properties of the sentence. They capture relevance to the query, as well as relatedness to domain-specific concepts. In our model, we rely on easy-to-compute features, which have proven successful in other retrieval tasks. All of the following features are extracted for each of the candidate sentences.

1. *Sentence Length.* The count of terms per sentence after removal of stop words.
2. *Sentence Location.* The relative position of the sentence within the abstract as it highlights the importance of a sentence. The feature is calculated as the location divided by the total number of sentences. However, instead of using the original location of the sentence, we use its position in the candidate set.
3. *Term Overlap.* This measures the number of terms that are found in both the query and the sentence, after removing of stopping words, and also stemming all terms using Porter stemmer [22].
4. *Synonyms Overlap.* The fraction of overlap between the query terms and sentence terms or their synonyms fetched from WordNet.
5. *BM25 score.* The Okapi BM25 framework is a Bag-of-Words model with a collection of scoring functions combined. For a query Q containing n terms $\{\mathrm{q_1}, \mathrm{q_2}, \mathrm{q_3}, ..., \mathrm{q_n}\}$ and a sentence length S, the similarity score between Q and S is calculated as

$$\text{BM25score}(S,Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i,S)\cdot(k_1+1)}{f(q_i,S)+k_1\cdot(1-b+b\cdot\frac{|S|}{\text{avgSl}})}$$

where IDF is the *inverse document frequency*, *avgSl* is the average sentence length and k-b are two tuning parameters set to 0.75 and 1.5 respectively in our implementation.

6. *Word Embeddings.* Cosine similarity between query and candidate sentences' word vectors pre-trained using a set of over 10 million PubMed abstracts.

The first four features are a subset of the features suggested and used by Metzler and Kanungos work on text summarization [18]. The last two features give more insight into the context of the sentences. In general, word embeddings are a vector representation of words co-occurrences that regard words as contexts, and hence gives a better apprehension of the word meanings. The vectors are generated from a large collection of data through Neural Networks. We take advantage of the word vectors, provided by the BioASQ challenge team [13], trained on a corpus of 10,876,004 English abstracts of biomedical articles from PubMed with 1,701,632 distinct words (types). The implementation of the above features was accomplished using *Summaryrank*[2], a reference package released for a similar task [7,30].

Sentence Ranking. As mentioned above, there might be multiple sentences that answer the query, but only the most suitable should be extracted. For example, while the next two phrases positively answer the question, only the second one is chosen as a claim. *In patients with hypertension, does treatment with ACE inhibitors, compared to placebo, reduce risk of cardiovascular event or improve blood pressure?*

– The vascular pathophysiologic alterations of ISH-a decreased aortic distensibility-can be improved with long-term lisinopril treatment, whereas values deteriorate further in placebo-treated subjects. [PMID: 11336102][assertion= YES]
– These results, in one of the first studies including subjects with previously untreated ISH only, indicate that lisinopril treatment might favorably influence the cardiovascular risk of ISH. [PMID: 11336102][assertion= YES]

Learning to Rank (LTR) is better suited for such a task since it differs from traditional machine learning techniques; the latter solves it as a classification problem while the aim is an optimal order of the instances in the list [17]. In our research, we evaluated two of the popular state-of-art learning to rank algorithms,*LambdaRank* and *LambdaMART*. LambdaRank is a succesor of RankNet that only uses the gradient of the costs instead of the model score. LambdaMART benefits from the strengths of MART, Multiple Additive Regression Trees, and LambdaRank by combining regression trees boosting, used in MART with a cost function derived from LambdaRank [5]. Our proposed model implements

[2] https://github.com/rmit-ir/SummaryRank.

a LambdaMART function, because it outperformed lambdaRank, with training metric NDCG@10. The Normalized Discounted Cumulative Gain (NDCG) is a cumulative measure of the ranking quality truncated at a particular rank level [15]. The model is trained on the generated feature vectors using the RankLib library[3] and the top ranked answer sentence is regarded as the output.

4.2 Contradiction Detection

The contradiction detection component is not regarded as a yes/no question answering system, but more as a semantic relation analyzer between two sentences. The system determines whether the input text has an entailment or contradictory relation.

Query Reformulation. This step aims at modifying the PICO-format question into a reduced list of keywords in a declarative form. In our approach, we consider each word in the question as a keyword unless it is a stop word, question word, or the substring "compared to placebo" is removed as it adds no value when identifying entailment or contradiction. Following that, we apply ClausIE [10], an open information extractor, to identify relations and corresponding arguments found in input question.

Features. Three features are used to identify the assertion values of claims.

Negation. The presence of negation is still the most effective feature of identifying oppositeness. Instead of relying only on the odd count of negation words in the sentence, our proposed model uses NegEx [6]. NegEx takes as input a keyword/concept and a sentence, and uses regular expressions and a predefined trigger word list to decide whether the concept is negated or affirmed. This module iterates three times over each question triple (*left argument, relation, right argument*).

Antonyms. The model includes direct and indirect antonyms; for two words w_1 and w_2, it checks if w_1 is an antonym of w_2 or an antonym of any of its synonyms and vice versa. Instead of comparing raw words, we use lemmas of words for better detection. However, even though the occurrence of antonym pairs in text is a direct and reliable indication of contradiction, it is limited by the low number of antonym pairs in current lexicons. Trying to overcome this limitation, we expand the antonym coverage by using two lexical resources, WordNet [19] and VerbOcean [8]. Below is an example that contains an antonym in *ManConCorpus*:

[3] https://sourceforge.net/p/lemur/wiki/RankLib/.

In women with pre-eclampsia, does treatment with L Arginine, compared to placebo, reduce blood pressure or pre-eclampsia

- L-Arginine load in pregnant women is associated with increased nitric oxide (NO) production and hypotension. [PMID: 10486782 - Assertion value: NO]

In this example, 'reduce' and 'increased' are not direct antonyms like 'good' and 'bad' but are still detected in model. This feature is computed as the count of antonyms per sentence.

Alignment. It is also important to include features that model text entailment. Alignment between sentences relies on mapping dependency graphs of two sentences with each other. The algorithm uses SpaCy[4] to generate dependencies, and a built-in similarity score is calculated for each word node in the query related to a similar one in the claim. Finally, the total alignment score is the sum of all output scores.

Classification. A linear support vector machine classifier is used to determine the relation of each input sentence, based on the output feature values. The model implements the classifier using the Scikit library [21].

5 Results

5.1 Claim Extraction Results

To evaluate the performance of the Learning to Rank framework and the efficiency of the features employed, we conduct two experiments. We first test the model using the first 5 features mentioned in Sect. 4.1 and then we repeat the test after adding the domain-based features covered by the word embedding trained on biomedical articles. For that purpose, we split the *ManConCorpus* into two sets for training and testing purposes. The training set consists of all abstracts with structured format, while the test set includes all unstructured abstracts. After the preprocessing phase, the candidate set has a total of 1212 and 339 sentences for training and testing, respectively. The test set includes 69 answers to only 15 of the 24 queries, while the training set covers all queries with 190 correct claims. Table 1 shows the performance results of the claim selection component. The authors in [1] relied on lexical similarity and a *Z-score* that computes the sentence relevance, with respect to the distribution of similarity scores of other sentences across the dataset. However, While this scoring function contributes to precision, it also affects the recall performance metric. The robustness of our proposed answer detection component relies on the combination of semantic and context features, with an effective ranking algorithm that ranks the sentences according to relevancy, instead of only classifying them as relevant/irrelevant.

[4] https://spacy.io/.

Table 1. Claim extraction results.

	Precision		Recall		F1	
	Answer	Non-answer	Answer	Non-answer	Answer	Non-answer
AlAmri [1]	0.56	0.92	0.57	0.92	0.56	0.92
Model (general features)	0.92	1	1	0.67	0.96	0.80
Model (general & domain-based features)	0.94	1	1	0.75	0.96	0.86

5.2 Contradiction Detection Results

Performance comparison between models is a non-trivial task, therefore we deploy the same evaluation metrics as in [3]. Since there is a bias in the distribution of *YES/NO* classes in the corpus, the results are best reported through precision, recall and F1. The baseline performance is measured by annotating all claims with the majority class *YES*. All evaluation results are shown in Table 2. Our model was able to improve the accuracy of detecting contradictions, namely the *NO* category, and still maintain good results regarding the entailment. The achieved improvement is due to the enhanced negation detection through the NegEx framework, and the inclusion of antonyms.

Table 2. Contradiction detection results.

	Precision		Recall		F1	
	Entailment	Contradiction	Entailment	Contradiction	Entailment	Contradiction
Baseline [3]	0.69	0.0	1.0	0	0.82	0.0
AlAmri [3]	0.85	0.80	0.94	0.60	0.89	0.69
Proposed model	0.95	0.85	0.93	0.89	0.94	0.87

6 Conclusions and Future Work

In this paper, we are interested in identifying conflicting findings reported in biomedical literature. We focus on information found in the abstracts as it summarizes all research methodology and conclusion, and conveys important findings without redundancy. It divides the extraction process into two phases, finding the relevant sentences and detecting contradiction. The model combines both semantic and domain-bases features, to enhance the claim detection process. It relies on an SVM classifier that integrates negation, antonyms and alignment scoring to detect conflicting statements. The evaluation results are very promising, specifically in the contradiction detection component, achieving better performance than other systems.

The results may be influenced by the small size of *ManConCorpus*, and hence further investigations are needed by scaling up the evaluation of the model on much larger corpora. Furthermore, a numeric mismatch in-between sentences is not regarded as a contradiction in the proposed system, since there are no contradiction instances of that type available in the corpus. However, when comparing clinical evidence found in biomedical literature, specifically when reporting recommended doses, considering variations in numeric values is important. Other possible extensions to the proposed model include incorporating domain knowledge resources, such as *UMLS*, Unified Medical Language System, and possibly integrating contrasting word embeddings [16].

References

1. Alamri, A., Stevenson, M.: Automatic detection of answers to research questions from medline abstracts. In: Proceedings of BioNLP, vol. 15, pp. 141–146 (2015)
2. Alamri, A., Stevensony, M.: Automatic identification of potentially contradictory claims to support systematic reviews. In: Proceedings of IEEE International Conference Bioinformatics and Biomedicine (BIBM), pp. 930–937, November 2015. https://doi.org/10.1109/BIBM.2015.7359808
3. Alamri, A.: The detection of contradictory claims in biomedical abstracts. Ph.D. thesis, University of Sheffield (2016)
4. Alamri, A., Stevenson, M.: A corpus of potentially contradictory research claims from cardiovascular research abstracts. J. Biomed. Semant. **7**(1), 36 (2016)
5. Burges, C.J.: From ranknet to lambdarank to lambdamart: an overview. Learning **11**(23–581), 81 (2010)
6. Chapman, W.W., Bridewell, W., Hanbury, P., Cooper, G.F., Buchanan, B.G.: A simple algorithm for identifying negated findings and diseases in discharge summaries. J. Biomed. Inform. **34**(5), 301–310 (2001)
7. Chen, R.C., Spina, D., Croft, W.B., Sanderson, M., Scholer, F.: Harnessing semantics for answer sentence retrieval. In: Proceedings of the Eighth Workshop on Exploiting Semantic Annotations in Information Retrieval, pp. 21–27. ACM (2015)
8. Chklovski, T., Pantel, P.: VerbOcean: mining the web for fine-grained semantic verb relations. In: EMNLP, vol. 4, pp. 33–40 (2004)
9. De Marneffe, M.C., Rafferty, A.N., Manning, C.D.: Finding contradictions in text. In: ACL, vol. 8, pp. 1039–1047 (2008)
10. Del Corro, L., Gemulla, R.: Clausie: clause-based open information extraction. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 355–366. ACM (2013)
11. Harabagiu, S., Hickl, A., Lacatusu, F.: Negation, contrast and contradiction in text processing. In: AAAI, vol. 6, pp. 755–762 (2006)
12. Ioannidis, J.P.: Why most published research findings are false. PLoS Med. **2**(8), e124 (2005)
13. Pavlopoulos, I., Aris Kosmopoulos, I.A.: Continuous space word vectors obtained by applying word2vec to abstracts of biomedical articles. Technical report, NLP Group, Department of Informatics, Athens University of Economics and Business, Greece Institute of Informatics and Telecommunications, NCRS Demokritos, Greece (2014)
14. Jameson, J.L., Longo, D.L.: Precision medicine – personalized, problematic, and promising. Obstet. Gynecol. Surv. **70**(10), 612–614 (2015)
15. Järvelin, K., Kekäläinen, J.: IR evaluation methods for retrieving highly relevant documents. In: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 41–48. ACM (2000)

16. Li, L., Qin, B., Liu, T.: Contradiction detection with contradiction-specific word embedding. Algorithms **10**(2), 59 (2017)
17. Liu, T.Y., et al.: Learning to rank for information retrieval. Found. Trends® Inf. Retrieval **3**(3), 225–331 (2009)
18. Metzler, D., Kanungo, T.: Machine learned sentence selection strategies for query-biased summarization. In: SIGIR Learning to Rank Workshop, pp. 40–47 (2008)
19. Miller, G.A.: Wordnet: a lexical database for English. Commun. ACM **38**(11), 39–41 (1995)
20. Padó, S., de Marneffe, M.C., MacCartney, B., Rafferty, A.N., Yeh, E., Manning, C.D.: Deciding entailment and contradiction with stochastic and edit distance-based alignment. In: TAC (2008)
21. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
22. Porter, M.F.: An algorithm for suffix stripping. Program **14**(3), 130–137 (1980). Google Scholar
23. Prasad, V., Cifu, A., Ioannidis, J.P.: Reversals of established medical practices: evidence to abandon ship. Jama **307**(1), 37–38 (2012)
24. Prasad, V., Vandross, A., Toomey, C., Cheung, M., Rho, J., Quinn, S., Chacko, S.J., Borkar, D., Gall, V., Selvaraj, S., et al.: A decade of reversal: an analysis of 146 contradicted medical practices. In: Mayo Clinic Proceedings, vol. 88, pp. 790–798. Elsevier (2013)
25. Preum, S.M., Mondol, A.S., Ma, M., Wang, H., Stankovic, J.A.: Preclude: conflict detection in textual health advice. In: 2017 IEEE International Conference on Pervasive Computing and Communications (PerCom), pp. 286–296. IEEE (2017)
26. Ritter, A., Downey, D., Soderland, S., Etzioni, O.: It's a contradiction–no, it's not: a case study using functional relations. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 11–20. Association for Computational Linguistics (2008)
27. Sarafraz, F.: Finding conflicting statements in the biomedical literature. Ph.D. thesis, University of Manchester (2012)
28. de Silva, N., Dou, D., Huang, J.: Discovering inconsistencies in pubmed abstracts through ontology-based information extraction. In: ACM Conference on Bioinformatics, Computational Biology, and Health Informatics (ACM BCB) (2017, p. to appear)
29. Sox, H.C., Greenfield, S.: Comparative effectiveness research: a report from the institute of medicine. Ann. Internal Med. **151**(3), 203–205 (2009)
30. Yang, L., Ai, Q., Spina, D., Chen, R.-C., Pang, L., Croft, W.B., Guo, J., Scholer, F.: Beyond factoid QA: effective methods for non-factoid answer sentence retrieval. In: Ferro, N., et al. (eds.) ECIR 2016. LNCS, vol. 9626, pp. 115–128. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30671-1_9

Following the Common Thread Through Word Hierarchies

Matthias J. Feiler(✉)

University of Zurich, Zürich, Switzerland
matthias.feiler@uzh.ch

Abstract. In this paper we develop a new algorithm for automatic taxonomy construction from a text corpus. In contrast to existing work, our objective is not to develop a general purpose lexicon or ontology but to identify the structure in a time–ordered sequence of documents. The idea is to identify "lead" words by which we are able to follow the common thread in the public discourse on a specific topic. Our taxonomy represents the backbone of the discourse (including names of protagonists and places) and may change over time. It is thus less rigid and universal than a lexicon and instead targets relationships that are valid in a given context. We present an example to illustrate the idea.

Keywords: Taxonomy learning · Topic tracking · On-line discourse

1 Introduction

Public attention to a topic has been shown to evolve in cycles [4]. Being able to determine the current phase within a cycle is useful both from an analytical and a practical viewpoint. Our motivation comes from finance where the maturity of a topic provides an indication of the extent to which relevant information has been *priced in*, i.e. reflected in the prices of traded assets. The concrete objective is to track the creation and evolution of themes in financial blogs. Blog conversations are by nature asynchronous and fragmented. The devices available in direct conversations for coordinating turns–of–talk [20] are not present. Participants need to track topical markers in order to follow the thread of a discussion. Our aim is to identify a temporary structure (a taxonomy) that supports the coherence of ideas and the emergence of a theme over many blogs.

One of the prerequisites for the formation of an over–arching stream of ideas is that blog participants are able to "connect the dots". Stories of universal truth relate to widely shared values or commonly understood situations. Being able to see the relation requires some degree of abstraction as the concrete format in which the story is told is unlikely to be identical over time. In a series of studies it has been shown that the human mind refers to a story in terms of a *schema* that contains abstract knowledge about a situation [9]. What are the word hierarchies that make certain schemas salient in the reader's mind? While fictional stories

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 149–158, 2018.
https://doi.org/10.1007/978-3-319-96136-1_13

often follow pre–defined scripts we are unlikely to find a “screenplay” in online–blogs. By contrast, the common thread seems to emerge spontaneously out of nothing. Moreover, blog communication is subject to social influence [23]: a path dependency arises if an initial exchange of ideas is deemed relevant by some bloggers which subsequently form an in–group of people sharing those ideas. Later entrants may be forced to adjust their own contributions in order to conform with the existing in–group. This significantly affects the universality of the learned taxonomy.

2 Related Work

A large body of literature exists on the problem of automatic taxonomy construction which can be broadly classified in heuristic, rule–based (see e.g. [7,18]) or statistical methods, e.g. [3,6]. The main objective is to generate a universally valid semantic lexicon complementing manual constructions such as WordNet [17]. The motivation behind taxonomies is to be able to leverage on an existing knowledge base through the principle of inheritance: the information structure of root words (hypernyms) is transferred to subordinates (hyponyms). On the other hand, it is widely agreed that semantic relations are not unique [14] which has led to the development of domain specific taxonomies that are constructed from scratch [15,24].

A common design principle is to start by extracting hypernyms from raw text e.g. by using a bootstrapping technique that starts with a root concept and a doubly anchored dependency pattern [16]. The learned concepts give rise to a (densely connected) network which is subsequently filtered and simplified to induce a taxonomy. For example Chu-Liu/Edmond’s algorithm may be used to find a spanning arborescence which, in turn, gives rise to a taxonomy based on the edge weights in the original graph [1,5]. In this work, the starting point is a co–occurrence matrix of terms computed over a domain–specific corpus. The motivation behind this choice comes conversation studies. The idea is that mutual understanding in human conversations is established “on–the–fly” through the creation of text worlds [8], i.e. shared mental representations of the situation at hand. We define text worlds as context–specific taxonomies; in fact, *local word hierarchy* would be a more suitable term for this temporary construction. The idea is to be able to follow a common thread in (public) discourse by identifying the hierarchy of keywords used in the conversation. Incidentally, higher–ranked words will correspond to more common (or abstract) ideas which brings this definition of a text world close to the notion of a taxonomy.

3 Taxonomy Generation

When people communicate, they rely on conventions in order to understand and produce meaning. Meaning is constructed in the mind of the listener using language as an input from which conceptual representations are formed. These linguistic inputs typically under–specify the concepts intended by the speaker and rely on the listener's ability to contribute the context needed to make a correct inference. In rational interaction models the speaker and listener apply (and expect) a common logic, or *cooperative principle* [11] to organize their speech acts.

The principle has been spelled out into four conversational maxims, the maxim of quality (truthfulness), quantity (informativeness), relevance and manner (conciseness). Mutual agreement on the maxims allow the speaker and listener to enrich an utterance by so–called *implicatures* which suggest an extension or modification of meaning beyond the literal interpretation, such as in *S1:* "Will he come?" *S2:* "His car broke down." which is decodable by *S1* into "He won't." by assuming that *S2* did not choose the answer if it was irrelevant. Also, *S2* supposes that *S1* has the background information that if cars brake down, people frequently do not manage to keep appointments. This is referred to as the *common ground* [2,22]. The interactive alignment model [19] emphasizes the importance of tacit coordination and *implicit* common ground. According to the model, grounding occurs automatically and the speakers' particular choices (i.e. which information to foreground) lead to an alignment of their (mental) representations.

Following a long tradition [21], conversation analysts study the way an interaction order [10] is established in practice, in particular how people take turns at talk, how they deal with overlaps and interruptions and how the sequence of utterances (and more general [speech] actions) is organized. Conversation analysis argues that the "...meaning of an action is heavily shaped by the sequence of actions from which it emerges, and that the social context is dynamically created [...] through the sequential organization of interaction", see [13], p.223. Any statement has to signal understanding of the preceding statements and prepare the floor for the next in order to establish coherence. This means that "each sentence [...] must contain some direct or indirect indication as to how it fits into the stream of talk", see [12], p.119. Two minds have to collaborate in order to "make progress" on the subject of their discussion. In the interactive alignment model this process occurs with a minimal amount of modeling what others know. According to the model, grounding occurs automatically through the speakers' particular choices i.e. which information to foreground. In this paper, we aim at identifying the words that have been foregrounded in a corpus of financial blogs.

3.1 An Algorithm for Taxonomy Extraction

Step 0 of our construction is to reduce and slice the corpus using a simple keyword filter and suitable time–intervals (e.g. a monthly grid). This generates a time–ordered sequence of sub–corpora containing documents related to a given area of interest. We represent every sub–corpus as a *bag of words* using a term–document matrix d. d is a $n \times m$ boolean matrix indicating the presence of a given term in a document where n is the number of terms (only *important* terms are included according to a global, i.e. topic–unspecific, tf–idf measure) and m is the number of documents in the sub–corpus. We aggregate over documents by setting $C = d\,d^T$ which is the co–occurrence matrix of terms in the sub–corpus.

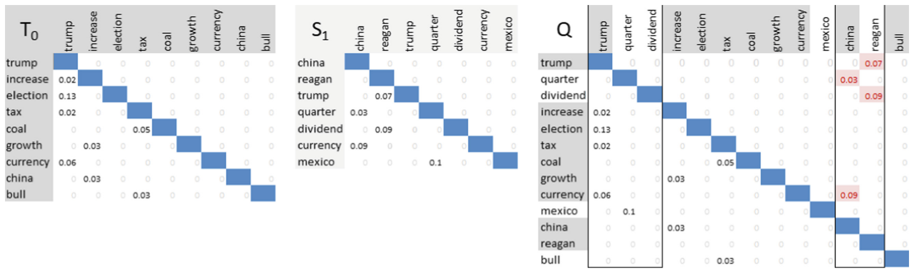
C forms the basis for the construction of our taxonomy T which proceeds in three steps:

1. Normalization of the rows of C enables us to interpret the entries of the co–occurrence matrix as intensities of a flow from every row term to the set of column terms in C. The resulting row–stochastic matrix A has a natural interpretation as the adjacency matrix of a directed graph representing the network flow originating at the term–nodes. Every edge in the resulting di–graph may be thought of as a reference that one term makes to another.
2. The nodes of the directed graph are rank–ordered by their in-degree (column sum) and the matrix A is re–arranged accordingly. We obtain: $P^T AP$ where P is the permutation matrix corresponding to the sort. Any high ranking node will be a parent node to the ones that reference it which means that the direction of any edge in the final structure is towards that node while edges out of the node are omitted. This means that the upper triangular matrix is set to zero and we obtain the intermediary result
$$T' = \mathrm{ltri}(P^T AP) \tag{1}$$
3. A *unique* parent is determined for every node by identifying the location of the maximum weight in every row of T'. We denote by $[\cdot]_{\max}$ the operator that sets all row entries except the maximum to zero and obtain
$$T = [T']_{\max} \tag{2}$$
corresponding to an in–tree or (anti–)arborescence representing a word hierarchy derived from the co–occurrence matrix. Notice that the maximum may not be unique as C is an integer matrix possibly containing duplicate entries which remain even after normalization. This is amended by choosing one of the solution candidates at random.

In summary, the number of references a term receives from others induces an order–relation which forms the backbone of our taxonomy T. In this paper, we are interested in studying how T evolves over a time–ordered sequence of sub–corpora. We introduce the index t referring to a point in the time–grid.

3.2 Taxonomy Evolution

Let T_t be a given taxonomy at instant of time t and let S_{t+1} be a new taxonomy created from the "next" sub–corpus i.e. from documents collected over the time interval $(t,\, t+1]$. Notice that S_{t+1} is an independently created taxonomy having no overlap with the previous step, i.e. the documents of T_t. The question is whether S_{t+1} may be attached to T_t in a natural way thereby extending the ideas of T_t to form a new, combined taxonomy T_{t+1}. Evolution in this context means that a given tree grows by forming branches which do not contradict the existing tree structure. While a simple keyword filter leads to an appropriate sub–corpus (for given a topic), the tree T_t specifies what is *commonly understood*



T0	trump	increase	election	tax	coal	growth	currency	china	bull
trump	🟦	0	0	0	0	0	0	0	0
increase	0.02	🟦	0	0	0	0	0	0	0
election	0.13	0	🟦	0	0	0	0	0	0
tax	0.02	0	0	🟦	0	0	0	0	0
coal	0	0	0	0.05	🟦	0	0	0	0
growth	0	0.03	0	0	0	🟦	0	0	0
currency	0.06	0	0	0	0	0	🟦	0	0
china	0	0.03	0	0	0	0	0	🟦	0
bull	0	0	0	0.03	0	0	0	0	🟦

S1	china	reagan	trump	quarter	dividend	currency	mexico
china	🟦	0	0	0	0	0	0
reagan	0	🟦	0	0	0	0	0
trump	0	0.07	🟦	0	0	0	0
quarter	0.03	0	0	🟦	0	0	0
dividend	0	0.09	0	0	🟦	0	0
currency	0.09	0	0	0	0	🟦	0
mexico	0	0	0	0.1	0	0	🟦

Q	trump	quarter	dividend	increase	election	tax	coal	growth	currency	mexico	china	reagan	bull
trump	🟦	0	0	0	0	0	0	0	0	0	0	0.07	0
quarter	0	🟦	0	0	0	0	0	0	0	0	0.03	0	0
dividend	0	0	🟦	0	0	0	0	0	0	0	0	0.09	0
increase	0.02	0	0	🟦	0	0	0	0	0	0	0	0	0
election	0.13	0	0	0	🟦	0	0	0	0	0	0	0	0
tax	0.02	0	0	0	0	🟦	0	0	0	0	0	0	0
coal	0	0	0	0	0	0.05	🟦	0	0	0	0	0	0
growth	0	0	0	0.03	0	0	0	🟦	0	0	0	0	0
currency	0.06	0	0	0	0	0	0	0	🟦	0	0.09	0	0
mexico	0	0.1	0	0	0	0	0	0	0	🟦	0	0	0
china	0	0	0	0.03	0	0	0	0	0	0	🟦	0	0
reagan	0	0	0	0	0	0	0	0	0	0	0	🟦	0
bull	0	0	0	0	0	0.03	0	0	0	0	0	0	🟦

Fig. 1. Attaching S_1 to T_0 through “zipping”.

by the topic. We ask how documents up to instant t prepare the ground for subsequent statements which either validate the word hierarchy in T_t or propose a new one. In the former case the attachment of S_{t+1} succeeds, otherwise a new tree is started. We define the attachment operator $h : T \times S \rightarrow T$ and obtain the evolution equation:

$$T_{t+1} = h(T_t, S_{t+1}, \theta) \tag{3}$$

Together with an initial condition T_0 this equation defines a path $\{T_t\}_{t \geq 0}$. The parameter θ refers to the minimum similarity among S_{t+1} and T_t such that S_{t+1} is attached, otherwise $T_{t+1} = T_t$. The operator $h(\cdot, \cdot)$ corresponds to the following construction:

Let V be the set of columns of S_{t+1} that also appear in T_t. We re–order V according to their ranking in T_t. This will create entries in the upper triangle of the combined matrix Q. We let ξ_{t+1} be the sum of these entries normalized by the sum over all elements in S_{t+1}. Parents (i.e. higher–ranking columns) in S_{t+1} that also appear in T_t may thus become children in Q, see Fig. 1 where $t = 0$. By contrast, all columns W that do not appear in T_t retain their ranking relative to the next higher–ranking column in V. In other words, parents in S_{t+1} take their children with them as long as this does not create a contradiction with existing parents in T_t. Columns in W are inserted together with corresponding rows to form an extended, quadratic matrix Q. If $\xi_{t+1} \leq \theta$, the similarity of S_1 and T_0 is sufficient for integration and the equivalent of step three (see Sect. 3.1) is repeated on ltri(Q) in order to determine unique parents for every term (except the highest–ranking one). More precisely, the $[\cdot]_{\max}$ operator is applied only on new terms (those in W), while existing child–parent relations (those in T_t) are retained. This means that if term A is parent to term B in T_t it will continue to be parent in the new taxonomy T_{t+1}. If $\xi_{t+1} > \theta$, the ranking of columns in S_{t+1} is deemed too different than the one in T_t which means that S_{t+1} cannot be attached. The overall procedure corresponds to the “zipping” of two trees.

This is illustrated in Fig. 1: The reordering of the column “china” (top rank in S_1) creates entries in upper triangle of Q, such as the reference coming from “currency” (value: 0.09). “Currency” is subordinate to “china” in S_1 but no longer in Q as the existing tree T_0 expects a higher rank of “currency” than “china”. By contrast, column “reagan” also has a lower rank than “china” in S_1 but it does not appear in T_0 which means that it can be moved to the combined structure Q together with "china". The references for "reagan" come from "trump" and "dividend" which are both higher–ranking in Q and will therefore be set to zero. Notice that the determination of the rank occurs *before* the actual tree construction, i.e. before setting the upper triangular matrix to zero and before determining unique parents. This means that high–ranked terms (such as "reagan") may end up with no references (if these come from even higher–ranking terms). It follows that the rank in the adjacency matrix T encodes some extra information about the taxonomy which is not reflected in the graph of T.

Comment: To some extent, the zipping operation mimics the coordination procedures in natural conversations: an existing taxonomy prepares the ground for future hierarchies to be attached (as branches). If this occurs, the top node of the sub–hierarchy is attached to a point in the taxonomy thus making a clear reference to its "origin". In dialogue, speaker and listener adapt to each other in the sense that messages are designed to the listener (gradually incorporating the listener's mental representations of the matter discussed) while listeners provide clear references to (or even repeat) what they heard. Over time, a chain of statement-response type of pairs (so–called adjacency pairs[1]) result which form the basis of the common thread in the dialogue. Our algorithm design draws on this basic mechanism to construct a taxonomy that evolves over time. It is clear that the analogy fails at the point where we do not consider individual conversation partners but merely aggregate text documents published over a given time period. However, if we allow ourselves to view the documents as statements of an abstract aggregate speaker or listener, the number of successful attachments indeed reflects the degree of *mutual understanding* that develops among the contributors to the text corpus.

4 Example

In Fig. 2 we illustrate the above ideas on a sub–corpus around the keyword "protectionism". The four graphs display snapshots of the evolution of the topic taxonomy. The underlying raw data was collected in monthly batches over a time–span from May 2015 to Mar 2018. The first graph is the result of a pure taxonomy extraction from the co-occurrence matrix C obtained in May 2015. In the subsequent steps new sub–graphs are attached using Eq. (3), thereby incrementally growing the initial tree. In this illustrative example, the threshold is set to $\theta > 1$ effectively posing no constraint to the attachment process. The number of terms in C is $n = 100$ and the number of documents behind

[1] Adjacency pairs constitute the central organizing format in natural conversations. They consist of two turns by two different speakers which are relatively ordered. The so–called "first pair part" initiates the exchange whereas the "second pair part" *responds* by providing a relevant follow–up statement. In this paper, we assume that the responses are always "pair–type related"; by starting with a filtered sub–corpus we exclude improper pairings whose dialogue–equivalent would roughly read: "Would you like some tea?"–"Hi!" [21].

C varies $m = 100 \dots 500$ depending on the intensity of the discussion around "protectionism" (i.e. the number of documents retrieved in a given month).

Notice that the study of the intensity evolution is outside the scope of this paper which is focused on the *qualitative* evolution of the topic. In fact, the entries of C (absolute occurrence counts) are normalized as part of the taxonomy extraction algorithm. In order to keep the presentation uncluttered only significant nodes are displayed in the graphs. We use an additional threshold $\theta_0 = 1.5$ for the column sum (corresponding to the aggregate endorsement received by the node) below which we do not display a node[2]. In the chart at the bottom of Fig. 2 we report the monthly values of the dissimilarity measure ξ defined above.

In May 2015, we find that our algorithm puts "trade" as a root together with qualifiers "global" and "china" which seems very close to a textbook (i.e. lexical) definition of protectionism. Around Sep 2016, near the pinnacle of the US electoral campaign, the discussion on trade has evolved to a more nuanced level containing specific issues such as "steel" and a number of macro–aspects such as "inflation rate". At the same time, a new subtree has emerged containing the "clinton/trump – scenario". Notice that the subtree has no visible connection with the protectionism discussion but has already been assigned a position within the hierarchy (see the comment at the end of Sect. 3.2). After the election, from Nov 2016 onwards, we notice an accentuated increase in the dissimilarity ξ. This marks a change in the perspective on "protectionism" which is reflected in a re–shuffle of the word–order developed thus far. In other words, new subtrees attached to the Sep 2016 tree generate more and more entries in the upper triangular matrix of the combined taxonomies. Referring to the above description of "zipping" we know that the attachment points are elements of the column set V which intersects the existing tree. The question is if these entries in the word hierarchy entail a sufficient number of sub–ordinates (i.e. elements of W having no overlap with existing structures) or even followers (i.e. sub–ordinates that also connect to terms in V). In such a case, ξ will decrease as no further contradictions are produced.

It is interesting to consider what kind of input would lead to a continuous high level of ξ: this would correspond to a sustained re–shuffling of the word order which would mean that the position of any new word introduced to the hierarchy would be revised in subsequent months. This is characteristic of a change in viewpoints or interpretations on a topic as can be seen in the period after Sep 2016. The Nov 2016 taxonomy shows that two subtrees may initially grow independently with "trump" becoming the root of the "election" tree. In Mar 2017 this tree finally connects to the "trade–china–rate" tree bringing a number of new elements into the discussion such as "mexico", "currency" and "dollar". It should be noted that the "trump" compound is sub–ordinate to the earlier discussion around the macro effects of "protectionism".

Notice also, that the structure of the final taxonomy depends on the initial condition: if the attachment process had been started at a later stage, say in Nov 2016, "trump" would have been the root. An important feature of the

[2] This level of θ_0 is thus 1.5 times the row sum in the normalized matrix C.

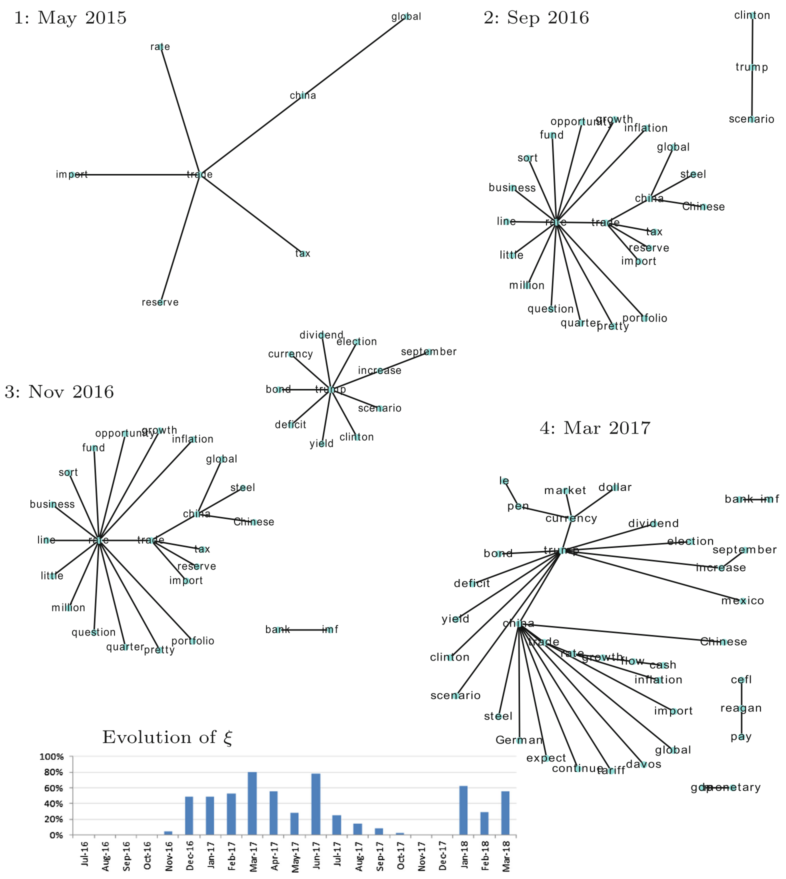


Fig. 2. Example: taxonomies of the theme "protectionism" (generated through attachment of monthly sub–corpora according to Eq. 3) and evolution of the dissimilarity measure ξ.

proposed technique is indeed that it indicates the origin of a discussion. In fact, our construction is path–dependent, as is the formation of common ground in natural conversations. After Mar 2017 we see that ξ declines indicating a steady state in the taxonomy. This temporary definition of the implications and ramifications of protectionism is again challenged in Jun 17 and Jan 18 as reflected by a resurgence of ξ.

5 Conclusion

The paper presented a new algorithm for the automatic construction of a temporary taxonomy used in on–line conversations to establish a (context–dependent) common ground. The taxonomy evolves as new sub–topics enter the conversation. A natural question about the algorithm is how it may be benchmarked.

Two taxonomies may be compared in terms of their "normative capacity", i.e. their ability to establish a word hierarchy which attracts followers (in terms of trees attached). Given a base tree, the question is whether subsequent trees may be attached without significant changes in the word order. If ξ in the above construction is large (and remains so), the trees contain the same set of keywords but in a different order. It is then possible to search for another base which leads to a decreasing ξ as new trees are attached. This is equivalent to a gradual specification of the defining word hierarchy associated with a topic. If ξ indeed decreases, more and more following trees attach to an existing base using the same word order and adding new words which do not contradict the existing structure. Notice the *self–referential* nature of this definition: a taxonomy is "true" if it is used by many subsequent documents. This is in contrast to benchmarking against an exogenous ground truth as given by a lexicon or an established ontology. In on–line discourse, "ground truth" is a fluid concept and reflects what most people think. The fact that a taxonomy is validated *from within* – through mutual understanding among contributors – marks a departure from standard problems in taxonomy construction.

References

1. Chu, Y.J.: On the shortest arborescence of a directed graph. Sci. Sin. **14**, 1396–1400 (1965)
2. Clark, H.H., Marshall, C.R.: Definite reference and mutual knowledge. Psycholinguistics: Crit. Concepts Psychol. **414** (2002)
3. Cohen, T., Widdows, D.: Empirical distributional semantics: methods and biomedical applications. J. Biomed. Inform. **42**(2), 390–405 (2009)
4. Downs, A.: Up and down with ecology-the issue-attention cycle. Public Interest **28**, 38–50 (1972)
5. Edmonds, J.: Optimum branchings. J. Res. Natl. Bureau Stan. B **71**(4), 233–240 (1967)
6. Fountain, T., Lapata, M.: Taxonomy induction using hierarchical random graphs. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 466–476. Association for Computational Linguistics (2012)
7. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In: IJCAI, vol. 7, pp. 1606–1611 (2007)
8. Gavins, J.: Text World Theory. Edinburgh University Press, Edinburgh (2007)
9. Gick, M.L., Holyoak, K.J.: Schema induction and analogical transfer. Cogn. Psychol. **15**(1), 1–38 (1983)
10. Goffman, E.: Forms of Talk. University of Pennsylvania Press, Philadelphia (1981)
11. Grice, H.P.: Logic and conversation, pp. 41–58 (1975)
12. Gumperz, J.J.: Mutual inferencing in conversation. In: Mutualities in Dialogue, pp. 101–123 (1995)
13. Heritage, J.: Conversation analysis and institutional talk. In: Handbook of Language and Social Interaction, pp. 103–147 (2005)
14. Hovy, E.: Comparing sets of semantic relations in ontologies. In: Green, R., Bean, C.A., Myaeng, S.H. (eds.) The Semantics of Relationships, vol. 3, pp. 91–110. Springer, Heidelberg (2002). https://doi.org/10.1007/978-94-017-0073-3_6

15. Kozareva, Z., Hovy, E.: A semi-supervised method to learn and construct taxonomies using the web. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 1110–1118. Association for Computational Linguistics (2010)
16. Kozareva, Z., Riloff, E., Hovy, E.H.: Semantic class learning from the web with hyponym pattern linkage graphs. In: ACL, vol. 8, pp. 1048–1056 (2008)
17. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: an on-line lexical database. Int. J. Lexicography **3**(4), 235–244 (1990)
18. Pantel, P., Pennacchiotti, M.: Espresso: leveraging generic patterns for automatically harvesting semantic relations. In: Proceedings of the 21st International Conference on Computational Linguistics, pp. 113–120. Association for Computational Linguistics (2006)
19. Pickering, M.J., Garrod, S.: Toward a mechanistic psychology of dialogue. Behav. Brain Sci. **27**(02), 169–190 (2004)
20. Sacks, H., Schegloff, E.A., Jefferson, G.: A simplest systematics for the organization of turn-taking for conversation. Language 696–735 (1974)
21. Schegloff, E.A.: Sequence Organization in Interaction: Volume 1: A Primer in Conversation Analysis, vol. 1. Cambridge University Press, Cambridge (2007)
22. Stalnaker, R.: Common ground. Linguist. Philos. **25**(5–6), 701–721 (2002)
23. Turner, J.C.: Social Influence. Thomson Brooks/Cole Publishing Co, Pacific Grove (1991)
24. Velardi, P., Faralli, S., Navigli, R.: Ontolearn reloaded: a graph-based algorithm for taxonomy induction. Comput. Linguist. **39**(3), 665–707 (2013)

Document Clustering Using Local and Universal Knowledge

Kazem Qazanfari(✉) and Abdou Youssef

The George Washington University, Washington, DC 20052, USA
kazemmit@gwu.edu

Abstract. In almost all real-world text clustering problems, the distribution of the repository samples and the real distribution of the clusters' concepts are rarely equivalent, which reduces the accuracy of the document clustering methods. Let *U(f)* and *L(f)* be the distribution functions of the extracted features based on *U*niversal knowledge and *L*ocal -repository- knowledge, respectively. Having the same distribution functions *U(f)* and *L(f)* is desirable; however, in real-world situations, these two distribution functions are not equal and they might be even quite different. In this paper, we show how the difference between these two distribution functions could decrease the accuracy of the document clustering algorithms. To address this issue, two different methods are proposed which combine information from the local and universal knowledge efficiently. In the first method, a special transform *T* is introduced to combine the similarities of each pair of documents derived from the local and the universal knowledge. In the second method, the local and the universal knowledge are combined, per document, by concatenating each document's feature vector derived from the local knowledge to the document feature vector derived from universal knowledge. The impact of the proposed methods on clustering is tested on two well-known datasets, Reuters and 20-Newsgroups. Experimental results show that by using either local or universal knowledge to generate the feature vectors, some documents could be assigned to a wrong cluster. However, we show that our proposed methods significantly improve the document clustering performance, thus demonstrating the benefit of enhancing local knowledge with universal knowledge in an efficient way.

Keywords: Text mining · Document clustering · Transfer learning

1 Introduction

With the recent increase in textual data, having an efficient and high-quality clustering algorithm to assign textual data to different unknown topics is needed for applications such as text mining, information retrieval, corpus summarization, social networks, news and politics, economics, target marketing, medical diagnosis [1]. Also, the textual data to be clustered could be documents, paragraphs or sentences.

The problem of document clustering is defined as follows. Given a set D of unlabeled documents, $D = \{D_1, D_2, \ldots, D_m\}$, a document clustering algorithm organizes documents into different groups, called clusters, where the documents in each cluster share some common properties according to some defined similarity measure. Clustering algorithms can generate either disjoint or overlapping clusters [2].

The problem of document clustering includes two major phases: Converting each document to a vector of features, and clustering of the documents using their feature vectors. The former phase is called feature extraction. Extracted features should preserve as much of the original document concepts as possible, while keeping the time and space complexity of the extraction process reasonable [3]. Generally, feature extraction from the collected documents include following steps:

- Document format normalization: Depending on how the documents were generated, there might be a variety of different formats such as Microsoft word, Html page and raw ASCII text. Therefore, the first step of feature extraction is to convert all documents to one format like raw ASCII text.
- Tokenization: In text mining, tokenization is the process of converting a sequence of characters into a sequence of tokens that have identified meaning.
- Stop words removal: Sometimes, some common words like "a", "an" and "the", called stop words, have little value in describing the meaning of the documents. Removing such words improve the speed and accuracy of clustering.
- Lemmatization and stemming: The next step is to convert the remaining tokens to a normalized form. This process is called stemming or lemmatization. The effect of this step is to reduce the number of distinct word types. For example, after stemming, two words "race" and "racing" will be considered to be "race" [4]. Depending on the application, this step might be helpful.
- Vector generation: The collective set of distinct tokens is typically called a dictionary which is used for creating the feature vector. The most general and widely used features in text mining techniques are unigram, bigram and more complex combination of tokens. Also, there are several representations for each token in a document such as binary, *tf*, *tf-idf*, binary *tf-idf, etc.*

The second phase of document clustering is to use the generated feature vectors to cluster documents. Many document clustering techniques have been proposed for text clustering. Generally, there are two major types of clustering techniques: partitional clustering and hierarchical clustering [5]. Partitional clustering methods, which divide the data points into a pre-defined number of clusters, include K-means [6], Bisecting K-Means [7], K-Medoids [8], KL and generalized I-divergence [9, 10]. Hierarchical methods create clusters either from singleton data points to a cluster or vice versa [11–13]; the former technique is known as agglomerative, and the latter is called divisive hierarchical clustering. Other clustering techniques have been proposed, such as those that use Gaussian Mixture Model (GMM) [14] or different types of neural networks [15–17].

Given a collection of m documents $D_1, D_2, \ldots, D_m$, and assuming that each feature vector is of some length n, each document D_i is represented by a feature vector denoted as:

$$D_i = \left(f_1^i, f_2^i, \ldots, f_n^i\right)^T$$

where f_j^i is the value of the j^{th} feature in document D_i. The j^{th} feature could be calculated simply by multiplying a variant of term frequency (*tf*) weight with a variant of inverse document frequency (*idf*) weight of the j^{th} term in the dictionary, or it could be calculated by a complicated deep neural network like FastText [17]. Tables 1 and 2 show the different variants of *tf* and *idf*.

Table 1. Different variants of term frequency (*tf*) weight

Weighting scheme	*tf* weight
Binary	$0, 1$
Raw count	$f_{t,d}$
Term frequency	$f_{t,d} / \sum_{t' \in d} f_{t',d}$
Log normalization	$1 + \log(f_{t,d})$
Double normalization 0.5	$0.5 + 0.5 . \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
Double normalization K	$K + (1 - K) . \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Table 2. Different variants of inverse document frequency (*idf*) weight

Weighting scheme	*idf* weight
Unary	1
Simple inverse document frequency	$\frac{n_t}{N}$
Inverse document frequency	$\log\left(\frac{n_t}{N}\right) = -\log\left(\frac{N}{n_t}\right)$
Inverse document frequency smooth	$\log\left(1 + \frac{N}{n_t}\right)$
Inverse document frequency max	$\log\left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t}\right)$
Probabilistic inverse document frequency	$\log\left(\frac{N - n_t}{n_t}\right)$

The problem of clustering could be formally defined as follows: Given m data point $X = \{x_1, x_2, \ldots, x_m\}$ in an n-dimensional real space R^n where each data point x_i is a feature vector $\left(f_1^i, f_2^i, \ldots, f_n^i\right)^T$ representing document D_i, determine K clusters $\{C_1, C_2, \ldots, C_K\}$ which each includes some data points from *X,* such that the sum of the "distances" of between the data points and the centers of their respective clusters is minimized. That is, the clustering problem is the following optimization problem:

$$\underset{C_1, C_2, \ldots, C_K}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{x_i \in C_k} \| x_i - \operatorname{center}(C_i) \| \quad (1)$$

where the norm $\|.\|$ is some arbitrary norm on R^n. The center of each cluster would represent the "overall meaning" of the documents in the cluster. Also, the above definition could be extended to have a variable number of clusters.

Since the content/concept of each document i is represented by the feature vector $D_i = \left(f_1^i, f_2^i, \ldots, f_n^i\right)^T$, the accuracy of each feature value might change the performance of the document clustering method. On the other hand, in a real-world document clustering problem, *L*ocal knowledge L (i.e. *repository data*) may not fully represent the characteristics of each cluster's meaning – in comparison to the *U*niversal knowledge U. Therefore, the distribution of the data in L for each cluster often does not match the real distribution of the cluster concept in U, and it causes to have inaccurate feature value. For example, consider w_i to be a word that appears in the local knowledge L. Let $U(w_i, C_k)$ and $L(w_i, C_k)$ be the relevance of word w_i to the concept of cluster C_k based on the universal knowledge U and local-knowledge L, respectively. In a document clustering problem, it is desirable to have $U(w_i, C_k) = L(w_i, C_k)$ or at least $U(w_i, C_k) \cong L(w_i, C_k)$. However, generally, the local knowledge L suffers from incomplete knowledge about the word w_i, which causes difference - sometimes quite significant - between $U(w_i, C_k)$ and $L(w_i, C_k)$.

Consider $L(w_i, C_k)$ to be the modeling of the local knowledge L about word w_i within cluster C_k. Specifically, we take $L(w_i, C_k)$ to be the likelihood (probability) of word w_i being in cluster C_k :

$$L(w_i, C_k) = \frac{f_{w_i, C_k}}{N_{C_k}} \tag{2}$$

where N_{C_k} is the number of documents in cluster C_k, and f_{w_i, C_k} is the number of documents in cluster C_k that have word w_i.

Consider Fig. 1, which illustrates the likelihood $L(w_i, C_k)$ of having word w_i in cluster C_k, for different clusters C_k, where

$$w_i \in \{Paint, Light, Design, Machine, Device, Starter, Across, Bottle, Complex\}$$

$$C_k \in \{\text{"comp.graphics" and "rec.autos"}\}$$

of the 20-Newsgroups dataset. Based on the general meaning of the above words, some words such as "paint" would relate more to computer graphics than to cars, other words like "machine", would relate more to cars than to computer graphics, and yet other words would be related equally well to both classes. Yet, the calculated likelihoods for these words based on the local knowledge show opposite statistics. Therefore, the accuracy of document clustering algorithms that use these words as features might be reduced if the values of these features are calculated based on just the local knowledge. In this paper, this phenomenon is called *inadequacy of knowledge*. The above-mentioned example shows this phenomenon for one-gram feature type; however, the same phenomenon is observed when bigrams or more longer combinations of words are used as features.

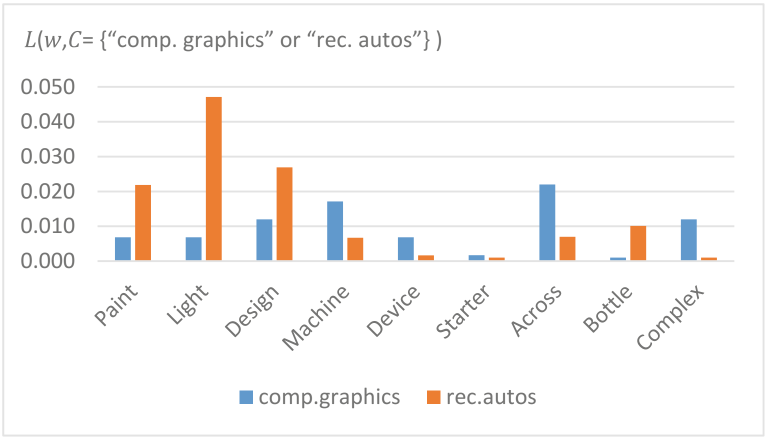


Fig. 1. Relevance of some words to two different clusters based on local knowledge20-Newsgroups

In this paper, to address the problem of inadequacy of knowledge, two bodies of knowledge are considered for document clustering: Universal knowledge, which is an external source of knowledge that has the general meanings of the documents, and the local knowledge, which has the domain-specific meaning of the documents for a given problem *P*. Then, these two bodies of knowledge are combined using two proposed methods. In the first method, a special transform *T* is introduced to combine the similarities of each pair of documents derived separately from the local knowledge and the universal knowledge. In the second method, the local and the universal knowledge are combined by concatenation of the two feature vectors derived from these bodies of knowledge. Then, using the combined local and universal knowledge, the documents are clustered using a clustering method. In the next sections, the details of the two proposed methods will be presented and their performance evaluated.

The rest of this paper is organized as follows. The proposed methods for combining the local and universal knowledge to improve the performance of clustering will be introduced in Sect. 2. Section 3 covers some experiments to cluster the documents of Reuters [18] and 20-Newsgroups [19] datasets, and shows that our proposed methods can significantly improve the performance of the document clustering methods. Finally, a summary of the achieved results and some suggestions for future directions will be presented in Sect. 4.

2 The Proposed Methods

If a body of knowledge *B* is viewed as universal knowledge, we denote it as *U*, and if it is the local knowledge (*repository dataset*), we denote it by *L*. In this section, two different methods are proposed to combine the benefits of these two bodies of knowledge *U* and *L* for document clustering purpose: (a) combining the similarities which are derived from *U* and *L* and (b) combining the feature vectors which are derived from the two bodies of knowledge.

2.1 First Method: Combining the Similarities

Given a problem P in some domain *"represented"* by a body of knowledge *(repository) L*, denote by $S_L(D_i, D_j)$ the similarity between two documents D_i and D_j based on the body of knowledge L. For example, S_L could be the Cosine measure (Eq. 3), which is a measure of similarity between two vectors in an inner-product space that measures the cosine of the angle between them:

$$S_L(D_i, D_j) = \frac{NR(D_i)\,.\,NR(D_j)}{\|NR(D_i)\|_2 \|NR(D_i)\|_2} \tag{3}$$

where $NR(D_i)$ and $NR(D_j)$ are D_i and $D_j^{\prime}s$ numerical feature vectors derived from the repository L, and $\|X\|_2$ is the Euclidean norm. Also, we view $S_U(D_i, D_j)$ as the similarity between documents $D_i\, and\, D_j$ based on the universal knowledge U that is independent of problem P.

In this section, a method will be proposed which combines $S_L(D_i, D_j)$ and $S_U(D_i, D_j)$ to create $S_{LU}(D_i, D_j)$ for each pair of documents D_i and D_j. The resulting $S_{LU}(D_i, D_j)$ models the overall similarity between the two documents based on both local knowledge L and universal knowledge U. To do so, a graph representation is used to visualize how $S_L(D_i, D_j)$ and $S_U(D_i, D_j)$ could be combined to create $S_{LU}(D_i, D_j)$.

Goal: The goal is to derive a fully connected graph $G_{LU}(V, E)$, where the nodes in V are the documents in the repository L, and each edge (D_i, D_j) between two documents D_i and D_j is given a weight $S_{LU}(D_i, D_j)$. These similarity values could later be used for clustering the documents.

Input: Two fully connected graphs $G_U(V, E)$ and $G_L(V, E)$.

- $G_U(V, E)$, is purely based on the universal knowledge U, where each edge $e = (D_i, D_j)$ is assigned a weight $S_U(D_i, D_j)$. The universal body of knowledge U is assumed to be modeled by the Pre-Trained Doc2Vec Model [20], which has been trained on Wikipedia dataset to generate the feature vector of each document. Therefore, $S_U(D_i, D_j)$ is the Cosine measure between the Doc2Vec vectors of D_i and D_j.
- $G_L(V, E)$ is purely based on the local knowledge L, where each edge $e = (D_i, D_j)$ is assigned a weight $S_L(D_i, D_j)$, taken to be the cosine between the D_i and $D_j^{\prime}s$ feature vectors derived relative to L. As the user usually has access to the local knowledge L, different kinds of feature vectors can be derived and utilized, such as conceptual features, contextual features (e.g., n-grams such as unigram and bigram), L-trained neural network features, as well as document-structure features and statistical

features like total number of words, number of sentences, and average length of sentences. In this paper, we use two different methods to represent the local knowledge L:

1. *tf-idf*, where the t^{th} feature of the vector for document D is:
2. $tfidf(D,t) = tf(D,t)\,.\,idf(D,t) = \frac{D^t}{D^{All}}\,.\,log\frac{n}{n^t}$ (4)

where D and t are the document and the contextual feature (e.g., unigram term), respectively, D^t is the number of times t appears in the document D, and D^{All} is the total number of terms in the document D, n is the total number of documents in L, and n^t is the number of documents in L that contain t.

3. *FastText* which is based on the skip-gram model [17], where each word is represented as a bag of character n-grams. This method associates a vector representation to each character n-gram; words being represented as the sum of these representations.

It is worth reiterating all the three graphs $G_{LU}(V,E), G_L(V,E)$ and $G_U(V,E)$ are fully connected graphs and have the same nodes.

Transform: Atransform function T is needed to combine the local knowledge graph $G_L(V,E)$ with the universal knowledge graph $G_U(V,E)$:

$$G_L(V,E) + G_U(V,E) \xrightarrow{T} G_{LU}(V,E) \tag{5}$$

$$\left(S_L\left(D_i,D_j\right), S_U\left(D_i,D_j\right)\right) \rightarrow S_{LU}\left(D_i,D_j\right) \tag{6}$$

where the details of the transform, specifically the value of $S_{LU}\left(D_i,D_j\right)$, will be provided later in this section. To illustrate visually the desired effect of the transform, consider an example of a clustering problem P with seven documents $V = \{D_1,D_2,D_3,D_4,D_5,D_6,D_7\}$ and two clusters C_1 and C_2. In a very simple scenario, suppose that based on $S_{LU}\left(D_i,D_j\right), C_1 = \{D_1,D_2,D_3\}$ and $C_2 = \{D_4,D_5,D_6,D_7\}$. The graph $G_L(V,E)$ is shown in Fig. 2. In this figure, the thickness of every edge $\left(D_i,D_j\right)$ is based on the similarity $S_L\left(D_i,D_j\right)$: the higher the similarity, the thicker the edge. Observe in Fig. 2 that there are some low-weight edges between some pairs of documents of the same cluster, such as (D_5,D_6), and that there are some high-weight edges between some documents from different clusters, such as (D_1,D_7). Such phenomena occur due to the *inadequacy of knowledge* (or, more accurately, *inadequate representation*) of L about problem P.

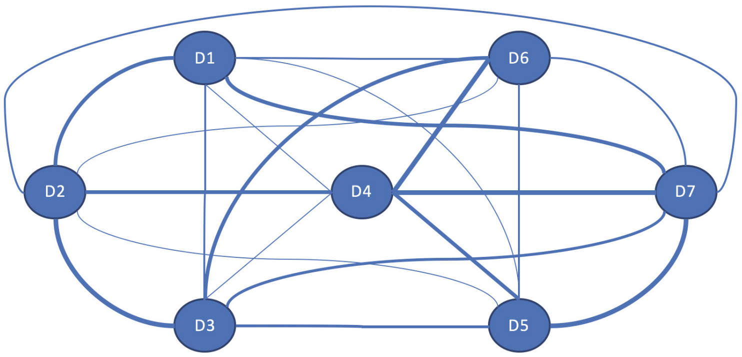


Fig. 2. A typical graph $G_L(V, E)$

By applying a good graph transform T on $G_L(V,E)$ and $G_U(V,E)$, the resulting graph $G_{LU}(V,E)$ will be like the graph in Fig. 3: the edge weights between the documents of the same cluster are high, while the edge weights between the documents from different clusters are low.

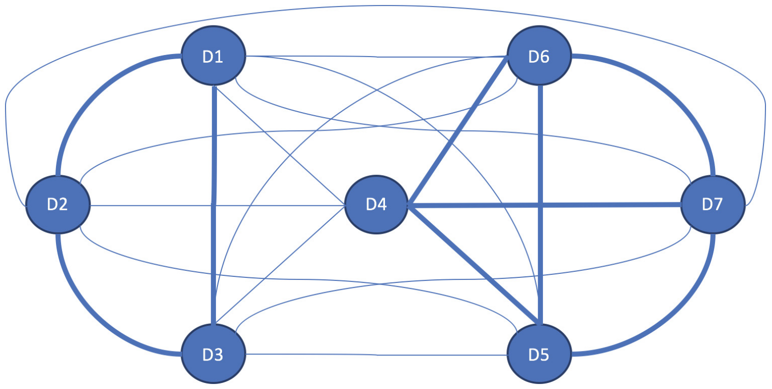


Fig. 3. A typical graph $G_{LU}(V, E)$ after transformation T

Before providing the graph transform, one term will be defined first, relative to a body of knowledge B, where B can be the universal knowledge U or the local knowledge L. Consider an edge $e = \left(D_i, D_j\right)$ between two documents D_i and D_j:

$$W_B^{norm}(e) = \frac{S_B(e)}{\max(all\, S_B(e_l)|e_l \in E)} \tag{7}$$

The term $W_B^{norm}(e)$ is the normalized similarity weight between two documents $D_i$and D_j relative to the body of knowledge B, so that it is always between 0 and 1.

In this paper, we propose and evaluate two different transforms presented in Eqs. 8 and 9:

$$S_{LU}(e) = \begin{cases} \alpha * W_L^{norm}(e) + (1-\alpha) * W_U^{norm}(e) & (8) \\ W_L^{norm}(e) \times W_U^{norm}(e) & (9) \end{cases}$$

In Eq. 8 α (and $1-\alpha$) is a weighting parameter between 0 and 1, optimized experimentally, relative to repository L. The first transform (Eq. 8) is called the *weighted-average transform*, and the second (Eq. 9) is called the *multiplicative transform*.

2.2 Second Method: Concatenating the Feature Vectors

Given two bodies of knowledge L and U, denote by F_L^D the feature vector of a document D derived from the local knowledge L, and F_U^D the feature vector of the document derived from the universal knowledge U. In the second method of combining the these bodies of knowledge, the two feature vectors F_L^D and F_U^D are concatenated to generate a comprehensive feature vector $F_{LU}^D = (F_L^D, F_U^D)$. The vector F_L^D could be any vector representation based on the local knowledge L such as *tf-idf* and *FastText*, and F_U^D is a Doc2Vec vector of length 300 produced by the Doc2Vec Model [20] that was trained on Wikipedia dataset.

3 Experiments and Discussion

3.1 Datasets

In our experiment, we use two corpora: Reuters [18] and 20 Newsgroups [19] datasets. Each of these two corpora will be treated our local knowledge. The 20 Newsgroups dataset is a collection of about 20,000 newsgroup documents, divided across 6 major different newsgroups covering a variety of topics such as computer, religion and politics. The Reuters dataset contains documents collected from the Reuters newswire in 1987. It is a standard text clustering benchmark and contains 21,578 samples in 135 categories. As a preprocessing step on the Reuters dataset, all categories that have less than 100 documents in the training set and the test set have been removed. The remaining dataset has 8210 documents in 20 categories.

3.2 Experiments

In the first experiment, we show how the proposed methods could cluster the samples that have been assigned to the wrong clusters by other methods including *tf-idf* and CNN (Doc2Vec). To help present the clustering performance, we created a small dataset including 8 short documents in two clusters, presented in Table 3. To generate feature vectors using the local and universal knowledge, we used *tf-idf* and CNN

(Doc2Vec) respectively. Then K-means algorithm and Cosine similarity were used to cluster these samples using different feature vectors including:

1. *tf-idf*
2. Doc2Vec
3. *tf-idf* + Doc2Vec (first proposed method)
4. *tf-idf* +Doc2Vec (second proposed method – using Eq. 8)

Table 3. Eight short document in two clusters

Text	Cluster label
The president of USA visited the president of Egypt in Cairo	Politics
Weather forecasting is a science to predict the conditions of the atmosphere	Weather
Political science, also called government, is a social science	Politics
Ten students visited the weather forecasting center QSA	Weather
Weather warnings are important forecasts because they are used to protect our life	Weather
The presidents of China and USA will sign an economy deal in summer	Politics
Ancient weather forecasting methods usually relied on observed patterns of events	Weather
QSA is an important center for education	Weather

Table 4. The result of clustering some samples by different feature generator methods

Clustering method	Resulting clusters
Clustering using *tf-idf*	Cluster #1: • The president of USA visited the president of Egypt in Cairo • The presidents of China and USA will sign an economy deal in summer Cluster #2: • Political science, also called government, is a social science • Weather forecasting is a science to predict the conditions of the atmosphere • Ten students visited the weather forecasting center QSA • Weather warnings are important forecasts because they are used to protect life and property • Ancient weather forecasting methods usually relied on observed patterns of events • QSA is an important center for education

(*continued*)

Table 4. (*continued*)

Clustering method	Resulting clusters
Clustering using CNN (Doc2Vec)	Cluster #1: • The president of USA visited the president of Egypt in Cairo • Political science, also called government, is a social science • The presidents of China and USA will sign an economy deal in summer • QSA is an important center for education Cluster #2: • Weather forecasting is a science to predict the conditions of the atmosphere • Ten students visited the weather forecasting center QSA • Weather warnings are important forecasts because they are used to protect life and property • Ancient weather forecasting methods usually relied on observed patterns of events
Clustering using tf-*idf* + CNN (Doc2Vec) (both proposed methods)	Cluster #1: • The president of USA visited the president of Egypt in Cairo • Political science, also called government, is a social science • The presidents of China and USA will sign an economy deal in summer Cluster #2: • Weather forecasting is a science to predict the conditions of the atmosphere • Ten students visited the weather forecasting center QSA • Weather warnings are important forecasts because they are used to protect life and property • Ancient weather forecasting methods usually relied on observed patterns of events • QSA is an important center for education

As shown in Table 4, since *tf-idf* only considers the statistics of the words instead of the meaning of them, the sample "Political science, also called government, is a social science" has been assigned to the wrong cluster based on the *tf-idf* feature vectors.

Also, there might be some sort of words (e.g., "QSA") in the repository that are not meaningful from the view point of CNN, yet they could be used for clustering purpose. Since CNN is not able to consider QSA in generating the feature vector, the sample "QSA is an important center for education" has been assigned to the wrong cluster based on the feature vectors that are generated by CNN.

However, since the proposed methods use the capabilities of both *tf-idf* and CNN, Table 4 shows that these samples have been assigned to the right clusters using both proposed methods.

In the next experiment, we find the optimized value for the parameter α of Eq. 8 experimentally for each dataset separately for when Doc2Vec and *tf-idf* are used for extracting the universal and local knowledge from documents. To do so, the performance of clustering methods is evaluated by the V-measure. The V-measure is the harmonic mean of homogeneity and completeness [21]:

$$V - measure = \frac{2 \times (\text{Homogeneity} \ \times \text{Completeness} \ )}{(\text{Homogeneity} + \text{Completeness} \ )} \tag{10}$$

Figure 4 shows the performance (V-measure) of the proposed weighted-average-transform based method (First method: Combining the similarities – using Eq. 8) for different values of α. As shown in this figure, the highest V-measure is achieved when $\alpha = 0.25$ and $\alpha = 0.4$ for 20-newsgroups and Reuters, respectively.

In the next experiment, the performance of the proposed methods will be compared with two different baseline clustering methods. To do so, we use either one or both unigram and bigram as the feature vectors and K-means as the clustering algorithm. Also, the performance of the proposed methods will be compared with FastText and Doc2Vec to generate the feature vectors based on the local knowledge and the universal knowledge respectively. Table 5 shows the performance of these methods and the proposed methods.

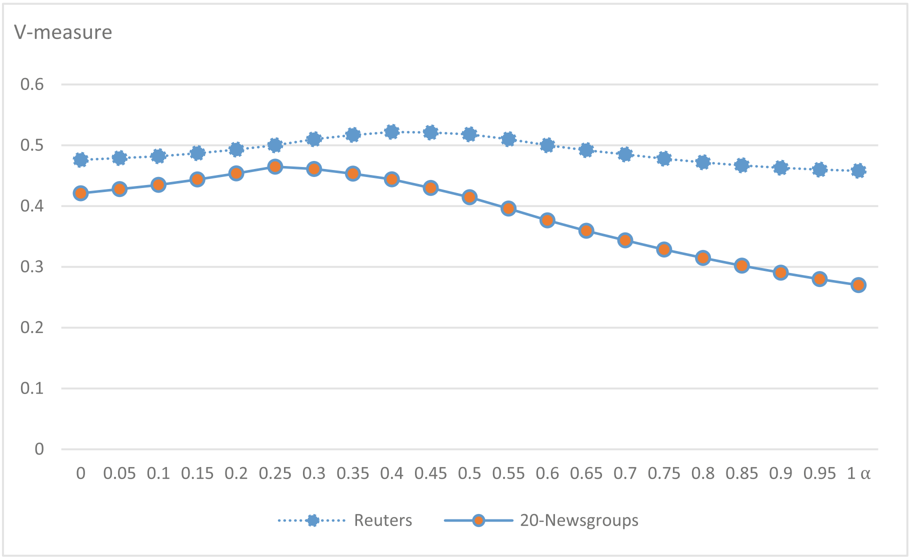


Fig. 4. The performance (V-measure) of the proposed weighted-average-transform based method (first method: combining the similarities – using Eq. 8) for different values of α

Table 5. Performance evaluation of some clustering algorithms on Reuters and 20-Newsgroups datasets

Method/feature set	V-measure	
	Reuters	20-newsgroups
Unigram + K-means	0.458	0.263
(Unigram, bigram) + K-means	0.462	0.295
CNN (Doc2Vec) + K-means	0.476	0.421
FastText + K-means	0.46	0.305
Proposed method#1 (*weighted-average transform on Doc2Vec and tf-idf*)	0.52	0.465
Proposed method#1 (*multiplicative transform on Doc2Vec and tf-idf*)	**0.522**	0.464
Proposed method#2 (*Concatenating the feature vectors of Doc2Vec and tf-idf*)	0.514	0.459
Proposed method#1 (*weighted-average transform on Doc2Vec and FastText*)	0.483	**0.481**
Proposed method#1 (*multiplicative transform on Doc2Vec and FastText*)	0.486	0.474
Proposed method#2 (*Concatenating the feature vectors of Doc2Vec and FastText*)	0.51	0.475

As shown in Table 5, among all mentioned methods, the first proposed method (based on the weighted-average- transform of Eq. 8, and on the multiplicative transform of Eq. 9) resulted in the highest V-measure for 20-Newsgroups and Reuters dataset, respectively. A close inspection of Table 5 shows that (1) both transforms of the proposed first method yielded similar performance, (2) the clustering performance resulting from the proposed method (of combining universal knowledge with local knowledge) is significantly superior to the one that utilizes only local knowledge. This demonstrates that the combining approach has considerable benefits for clustering performance.

4 Summary and Future Work

In almost all real-world text clustering problems, the repository knowledge is not comprehensive about the clusters' concepts; therefore, the clusters are built using incomplete information. In this paper, to address this issue, two methods for increasing the knowledge used in clustering were introduced.

To do so, we combined information from two bodies of knowledge, universal knowledge and local (repository) knowledge, using two completely different methods. In the first proposed method, the similarity between two documents is calculated by combining the similarities between them derived from the two bodies of knowledge. The combining of the similarities was done either as a weighted sum or multiplicatively. The other combining method simply concatenates the "local" and "universal" feature vectors of each document into a new feature vector. The performance of the proposed methods was evaluated on Reuters and 20-Newsgroups datasets. Experimental results show that by using either local or universal knowledge to generate the feature vectors, some documents could be assigned to the wrong cluster. However, using the proposed combining methods, those samples are assigned to the right cluster, and generally the clustering performance improved significantly.

In conclusion, our results demonstrate that efficient combining of information from local and universal body of knowledge has considerable advantage, at least for clustering performance, over the mere use of local knowledge.

As part of our future research, we will explore alternative ways of using universal knowledge to improve document clustering performance. Also, to generalize our proposed methods for using different sources of knowledge, we will focus in the future research on transfer learning or inductive transfer to use the stored knowledge while solving one problem and applying it to a different but related problem.

References

1. Berkhin, P.: A survey of clustering data mining techniques. Group. Multidimens. Data 25–71 (2006). https://doi.org/10.1007/3-540-28349-8_2
2. Tan, P.N., Michael, S., Vipin, K.: Data mining cluster analysis: basic concepts and algorithms. Introd. Data Min. **8**, 487–568 (2006)
3. Qazanfari, K., Youssef, A.: Contextual feature weighting using knowledge beyond the repository knowledge. Int. J. Comput. Commun. Eng. (IJCCE) (2018)
4. Qazanfari, K., Youssef, A., Keane, K., Nelson, J.: A novel recommendation system to match college events and groups to students. AIAAT **261**, 1–15 (2017)
5. Fahad, S.K.A., Wael, M.S.Y.: Review on semantic document clustering. Int. J. Contemp. Comput. Res. **1**(1), 14–30 (2017)
6. Singh, J.P., Nizar, B.: Proportional data clustering using K-means algorithm: a comparison of different distances. In: 2017 IEEE International Conference on Industrial Technology (ICIT), pp. 1048–1052. IEEE (2017). https://doi.org/10.1109/icit.2017.7915506
7. Forgy, E.C.: Analysis of multivariate data: efficiency versus interpretability of classification. Biometrics **21**, 768–780 (1965)
8. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data. An Introduction to Cluster Analysis, vol. 344. Wiley, Hoboken (2009)
9. Strehl, A., Ghosh, J., Mooney, R.: Impact of similarity measures on web-page clustering. In: Workshop on Artificial Intelligence for Web Search (AAAI 2000), pp. 58–64 (2000)
10. Chim, H., Deng, X.: A new suffix tree similarity measure for document clustering. In: 16th International Conference on World Wide Web, pp. 121–130. ACM (2007). https://doi.org/10.1145/1242572.1242590
11. Gower, J.C., Roos, G.J.S.: Minimum spanning trees and single linkage cluster analysis. J. R. Stat. Soc. Ser. C (Appl. Stat.) **18**, 54–64 (1969). https://doi.org/10.2307/2346439
12. Fisher, D.H.: Knowledge acquisition via incremental conceptual clustering. Mach. Learn. **2**, 139–172 (1987). https://doi.org/10.1007/BF00114265
13. King, B.: Step-wise clustering procedures. J. Am. Stat. Assoc. **62**, 86–101 (1967)
14. Liu, X., Gong, Y., Xu, W., Zu, S.: Document clustering with cluster refinement and model selection capabilities. In: 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 191–198 (2002). https://doi.org/10.1145/564376.564411

15. Xu, J., Xu, B., Wang, P., Zheng, S., Tian, G., Zhao, J.: Self-taught convolutional neural networks for short text clustering. Neural Netw. **88**, 22–31 (2017). https://doi.org/10.1016/j.neunet.2016.12.008
16. Gallant, S.I.: Method for document retrieval and for word sense disambiguation using neural networks U.S. Patent No. 5,317,507. 31 (1994)
17. Piotr, B., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
18. Lewis, D.D.: Reuters-21578, Distribution 1.0 (1987)
19. Joachims, T.: A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization, Computer Science Technical Report CMU-CS-96–118. Carnegie Mellon University (1996)
20. Jey, H.L., Timothy, B.: An empirical evaluation of doc2vec with practical insights into document embedding generation. arXiv preprint arXiv:1607.05368 (2016)
21. Rosenberg, A., Julia, H.: V-measure: a conditional entropy-based external cluster evaluation measure. In: EMNLP-CoNLL (2007)

Tweet Classification Using Sentiment Analysis Features and TF-IDF Weighting for Improved Flu Trend Detection

Ali Alessa and Miad Faezipour(✉)

School of Engineering, University of Bridgeport, Bridgeport, CT 06604, USA
aalessa@my.bridgeport.edu, mfaezipo@bridgeport.edu

Abstract. Social Networking Sites (SNS) such as Twitter are widely used by users of diverse ages. The rate of the data in SNS has made it become an efficient resource for real-time analysis. Thus, SNS data can effectively be used to track disease outbreaks and provide necessary warnings earlier than official agencies such as the American Center of Disease Control and Prevention. In this study, we show that sentiment analysis features and weighting techniques such as Term Frequency-Inverse Document Frequency (TF-IDF) can improve the accuracy of flu tweet classification. Various machine learning algorithms were evaluated to classify tweets to either flu-related or unrelated and then adopt the one with better accuracy. The results show that the proposed approach is useful for flu disease surveillance models/systems.

Keywords: Influenza · Machine learning · Sentiment
Social networking site · TF-IDF

1 Introduction

Social networking sites (SNS) are tools designed to provide users a space for social interactions. The content of social networking sites is produced by users. It includes huge data about users, shared thoughts and ideas, and real-time data of users' conversations and statuses. The rate of the data in SNS besides the growth of SNS's users, presents the important role of SNS's in real time analysis and predictions in many areas [1,2]. The areas include traffic [3–6], disaster prediction [7–11], management [12–14], networking [15,16], news [17–21] and many more. In the public health area, SNS provides an efficient resource for disease surveillance and also an efficient way to communicate to prevent disease outbreaks [22].

Traditional disease surveillance systems depend on official statistics based on patient visits to produce outbreak reports [23]. In the US, these reports are produced by the Center of Disease Control and Prevention (CDC) to inform healthcare providers about disease outbreaks or to be notified about the flu season. CDC publishes flu-related reports using the United States Influenza Like

Illness Surveillance Network (ILINet) that gathers flu-related information of outpatients from hundreds of healthcare providers around the states. ILINet shows accurate results in detecting flu outbreaks, but it is costly and takes a long time to issue the required reports. Therefore, different data from different resources have been used for surveillance. Examples include volumes of telephone calls, over-the-counter drug sales [23], search engine logs [24–29] and social networking data that can be used to get real time analysis for better services [30].

Comparing the different resources that have been used for surveillance such as search engine logs, SNS data is more descriptive and available for the public. It also has more demographic data and specific details. Therefore, it appears that SNS data is one of the best among the other resources for surveillance as well as a basis to simulate the spread of a disease with temporal analysis.

The SNS used in this study is Twitter Microblog because it is the most widely used social networking site. It is an efficient resource to track trends for several reasons. First, the high frequency of posted messages helps to perform minute-by-minute analysis. Second, Twitter provides detailed information about users such as demographic data. Third, users of Twitter are of diverse ages. Not only young people, but also middle aged, as well as technology savvy older population use this SNS [31].

In this paper, we present a machine learning based approach that predicts influenza trends in Twitter SNS. The approach includes preprocessing, feature extraction and classification. The preprocessing phase includes stemming and removal of stop words and bad characters. Then, the preprocessed data is used to extract features to be passed to a tweet classifier to distinguish between flu-related tweets and unrelated ones. The classifier results can then be used for further analysis for better flu trend prediction.

The main contributions of this study can be summarized as follows: (i) we show that TF-IDF weighting can improve the accuracy of tweet classifications; (ii) we show that considering sentiment analysis of the analyzed posts as a feature can improve the accuracy of the classification results; (iii) we show an evaluation of various machine learning algorithms for flu-related tweets and; (iv) we review the existing machine learning based frameworks for flu detection using the data of social networking sites.

The rest of the paper is organized as follows. The Related Work Section first presents the previous related work that use machine learning methods for flu tweet classifications. The Methodology Section, then, demonstrates the proposed approach for this study including preprocessing, feature extraction, training, and testing. The Discussion and Results Section presents a discussion and comparison between the proposed approach and the existing ones that use machine learning for flu tweet classifications. Finally, concluding remarks appear in the Conclusions Section.

2 Related Work

Most of the previous work on Twitter-based flu surveillance included machine learning methods to filter unrelated flu posts. A selected classifier is trained with an annotated dataset using a set of features. It has been found in the literature that different detection and prediction models utilized different classification methods with different feature extraction techniques.

Broniatowski et al. [32] and Lamb et al. [33] proposed a multi-level classification model that includes a binary classifier to distinguish between flu-related and unrelated flu tweets. The pre-classifiers are used to filter unwanted posts such as health-irrelevant posts in order to increase the efficiency of the flu-related/unrelated classifier in further stages/levels. It has been shown that multi-level classification can improve the classification accuracy.

Aramaki et al. [34] proposed a framework that consisted of two parts: a tweet crawler and a Support Vector Machine (SVM)-based classifier that was used to extract only the actual influenza tweets and excluded the unrelated ones such as news and questions. The initial dataset for this study was collected from Nov 2008 to June 2010. It included 300 million general tweets. Then, this dataset was filtered using the "Influenza" keyword to get a set of only flu-related tweets which contained 400,000 tweets. The flu-related dataset was divided into two parts: a training dataset which contained 5,000 tweets (November 2008) and a test dataset which contained all the remaining tweets from Dec 2008 to June 2010. The training dataset was assigned to a human annotator to label each tweet for being either positive or negative. A tweet is labeled positive if it met two conditions. First, the flu tweet should be about the person who posted the tweet or about another person in a nearby area (maximum an area of the city). If the distance is unknown, the tweet is considered negative. Second, the flu tweet should be an affirmative sentence and in a present tense or past tense with maximum period of 24 h which can be checked using specific keywords such as "yesterday". The SVM classifier was implemented using the Bag-of-Words feature representation. The authors compared the accuracy of the SVM-based classifier with 6 other different machine learning methods and found that SVM was the most accurate method.

Santos and Matos [35] also applied SVM-based classification to detect flu-like illness in Portugal using Twitter posts. For the purpose of training and testing, a dataset with 2,704 posts was manually annotated with 650 textual features. A subset of the annotated dataset was used to train the classifier. The classified tweets together with search queries were applied to a regression model as predictors. The classifier was implemented using the Bag-of-Words feature representation and the feature selection process was based on a Mutual Information (MI) value which is used to pick the best set of features. Each feature is applied to a true class and then the MI value is assigned to the feature. The value of MI is based on how the feature is related to the true class. A feature with high MI value represents being more related to the true class.

Yang et al. [36] proposed an SVM-based method to predict flu trends from Chinese social networking sites in Beijing. The authors claimed that this was the first study to predict flu trend from Chinese social networking sites. The collected data for this study included 3,505,110 posts from Sep. 2013 to Dec. 2013. Among those, 5,000 random posts were selected for manual annotation (sick and not sick labels) to be used for training and testing purposes. 285 of the sick posts and 285 of the not sick posts were picked for training. For higher accuracy, word based features were used instead of character based features. In addition, the term frequency-inverse document frequency (TF-IDF) method was considered for weighting. Different classifiers were compared to decide the best for the problem. The authors found that SVM was the best for big data problems.

Byrd et al. [37] proposed a framework based on the Naïve Bayes classifier. The framework consisted of several steps including preprocessing and flu tweet classification based on sentiment analysis. Three machine learning algorithms were evaluated and it was found that the highest accuracy method was the Naïve Bayes classifier. The Naïve Bayes classifier was implemented using the Stanford-CoreNLP (Natural Language Processing software) and trained using the OpenNLP training dataset which includes 100 annotated tweets. The sentiment analysis is considered accurate when there is matching between the predicted sentiment polarity with the manual assigned opinion of the sentiment. The authors found that Naïve Bayes was the most accurate one with 70% matching.

3 Methodology

In this study, we first collect the tweets of two labeled datasets and merge them together to be used as a training set. After that, the data is cleaned up by removing bad characters, digits and stop words. We then perform sentiment analysis by finding and analyzing the polarity score. The tweets are then stemmed and important features are extracted. We then train a classification model and finally, test and evaluate the model.

3.1 Corpus

For the training and testing dataset, we prepared a labeled dataset that is a combination of multiple manually labeled datasets obtained from [33,38]. This yields 10,592 tweets (5,249 flu-relevant and 5,343 flu-irrelevant posts) for the total dataset. Due to Twitter guidelines, the tweets in the obtained datasets were released with tweet IDs instead of the text of the tweets. Therefore, we developed a script that works together with the Twitter API to retrieve the corresponding tweet texts using the given IDs.

Twitter Influenza Surveillance Dataset. The labeled dataset obtained from [33] was initially filtered to contain any posts that have flu-related keywords. Then, every post in the dataset was labeled manually using Amazon Mechanical Turk: a service for tasks that require human intelligence [39]. It was prepared to train and test three flu-related classifiers that were used as a part of an algorithm for seasonal flu predictions. The dataset is, thus, divided into three sets, one for each classifier. The first set consists of tweets that were labeled as either flu-related tweets or not. The second one has tweets with labels of flu infections or flu awareness. The tweets in the last set were labeled as either the flu tweet being about the author or about someone else. For our dataset, we consider the tweets in the second and third datasets as flu-related tweets and combine all of them with only 2 labels: flu-related or unrelated.

Sanders Dataset. The labeled dataset obtained from [38] was prepared manually to train and test sentiment analysis algorithms. Each record in the dataset is annotated with a sentiment label, indicating a feeling toward either Google, Twitter, Microsoft or Apple. The labels are: positive, neutral, negative, and irrelevant. Since this dataset was prepared for sentiment analysis of topics that are not related to flu, we used all the tweets in this dataset except the ones with irrelevant labels as flu-unrelated tweets.

3.2 Preprocessing

Stop-words, punctuations and symbols were removed before the training and testing processes using the Natural Language Processing Toolkit (NLTK) [40]. Stop words such as “the” or “are” words are very frequent and may lead to inaccurate classification results if used as features. The preprocessing also includes stemming that is used to reduce words to their roots. There are many stemming algorithms available to use. For this study, Porter stemming which is one of the most commonly used stemming algorithms, is employed. It is a rule-based algorithm with five steps that is designed based on the idea that English suffixes are made of smaller and simpler ones. A suffix is removed if a rule in the five steps pass the conditions and is then accepted [41]. Figure 1 shows the overall preprocessing steps.

URL’s, Hashtags, and Mentions were kept in the corpus. They can be used as features for classification. URL’s were replaced with the keyword (url), and Mentions were replaced with the keyword (mn) to be used as one feature for classification.

3.3 Feature Extraction

In machine learning, a maximum accuracy can be achieved by selecting the best set of features. Therefore, feature selection is a crucial process in any classification problem. In text classification, the set of features is a subset of words that can be used to distinguish between different classes [42]. The selected words should provide useful information to be used for classification purposes. Thus, it is important to consider different techniques to convert the text in a way that can be processed to gain the required information. In this study, the used features are weighted term-based features, sentiment-based features, sylometric features, and flu-related keyword features.

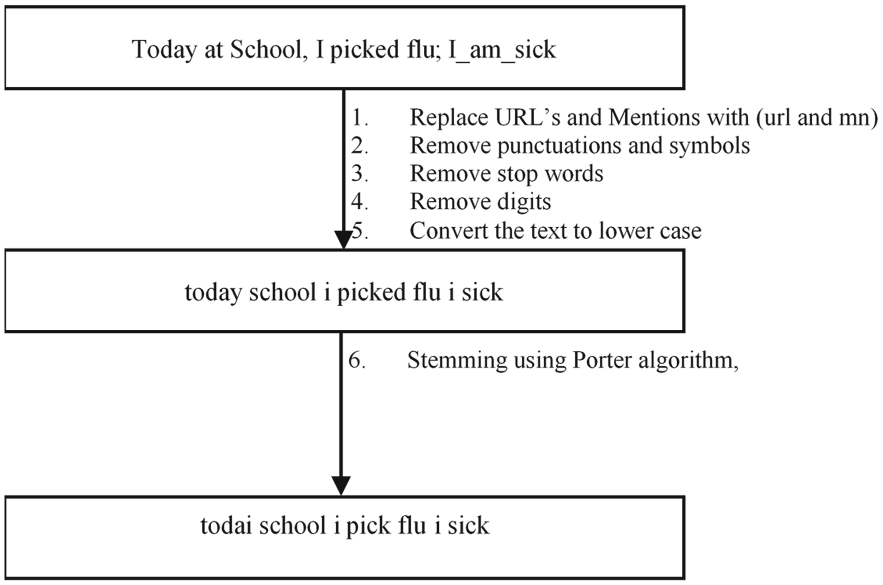

Today at School, I picked flu; I_am_sick
1. Replace URL's and Mentions with (url and mn)
2. Remove punctuations and symbols
3. Remove stop words
4. Remove digits
5. Convert the text to lower case
today school i picked flu i sick
6. Stemming using Porter algorithm,
todai school i pick flu i sick

Fig. 1. Text preprocessing

Textual Features Based on Term Frequency-Inverse Document Frequency (TF-IDF) Weighting. The basic technique in text classification is a direct word count, which breaks down a text into words and then counts every single word in the corpus, even the un-informative ones that may yield inaccurate results. Therefore, it is important to use smarter techniques. One of these techniques is the word/term weighting technique, which weighs the count for every word/term in the text. There are different techniques of word weighting which include Boolean weighting, Term Frequency weighting (TF), Inverse Document Frequency weighting (IDF) and Term Frequency-Inverse Document Frequency weighting (TF-IDF). Among the four types of word weighting techniques, only the IDF and TF-IDF techniques consider the importance of a word/term in the entire corpus instead of the importance of the word/term in only a document. It has been shown in [36] that TF-IDF is more accurate than IDF. Therefore, in this study, we used TF-IDF to weigh the text-based features that are extracted by breaking down the tweets into single words (uni-grams), terms composed of two words (bi-grams), and terms composed of three words (tri-grams).

TF-IDF value is obtained by multiplying the value of the Term-Frequency value by the value of Inverse Document-Frequency (Eq. 1). TF is the ratio between the term t with frequency n_t in a given document d and the total numbers of terms n in the document d (Eq. 2). IDF is the inverse of the number of documents that has the term t at least once. It is calculated using Eq. 3, which is the ratio between the frequency N_d of the documents d that have term t, and the total number N of documents d in the analyzed corpus.

$$TF\text{-}IDF(t,d) = TF(t,d) \times IDF(t) \tag{1}$$

$$TF(t,d) = \frac{n_t}{n} \tag{2}$$

$$IDF(t) = \frac{N_d}{N} \tag{3}$$

Stylometric Features. Retweets (RT), Mentions, and URL links were kept in the corpus. We included them to be used as features. URL links and Mentions to others were preprocessed by replacing them with url and mn keywords.

Topic-Related-Keywords Based Features. It is common to use seed words in text classification. For example, in sentiment analysis, a list of words, including nice and good, is used for positive sentiment and another list of words, including bad and poor, is used for negative sentiment. In this study, a set of flu-related keywords/terms were used as a set of features for flu-related tweets. The list includes some important influenza-related keywords, symptoms and treatments. The list of the keywords is kept in an array and then each tweet is checked against these keywords. If a keyword is found in the text of the tweet, a corresponding counter is increased. The counter value is weighted by dividing it by the tweet length. The final weighted value is used as a feature for that tweet. If no keywords are found in the tweet, a zero counter value is used as a feature.

Sentiment Based Features. Sentiment analysis is the process of extracting the sentiment of a text using contextual polarity. It is commonly used in classifying reviews of different products in the internet such as the sentiment of movies. In this study, we used TextBlob library to assign a sentiment to each tweet [43]. The TextBlob is a Python library that is used to analyze textual data. Based on the polarity score of a tweet, a sentiment value is assigned to the text: positive or negative.

3.4 Training and Testing

For training and testing, several supervised classification methods were evaluated to determine one with better classification accuracy. The evaluated classifiers include Random Forest, Naïve Bayes, SVM, Random Tree, J48, Bagging, K-nearest neighbors classifier using the Instance Based learning algorithm (IBK), Voting, and AdaBoost. The preprocessed labeled dataset was used to train and test the model of different classifiers using 10-cross validation as the experimental setting. The 10-cross validation is a method to validate the studied/built model by iterating through the labeled data 10 times with different subsets of training and testing for each iteration.

3.5 Performance Metrics

In this section, we present the performance of the classifiers using different metrics: accuracy, precision, recall, and F-measure. These metrics are used to provide a better overview of the model performance. The accuracy measure by itself is not a perfect measure if the dataset is not balanced. Precision and recall are better measures in the case of imbalanced datasets. The selected metrics can be computed using true positive (TP), true negative (TN), false positive (FP), and false negative (FN) measures, where TP refers to the rate of correctly classified instances as positive, TN refers to the rate of correctly classified instances as negative, FP refers to the rate of incorrectly classified instances as positive, and FN refers to the rate of incorrectly classified instances as negative.

Accuracy. It is a measure to evaluate the performance of a prediction model. It is the rate of the correctly classified labels. It is calculated by using Eq. 4:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

Precision. It measures the true positive predictions. The precision of a model is calculated by using Eq. 5:

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Recall. This measure is a sensitivity measure. It is used to evaluate a model's performance in predicting positive labels. It is calculated by using Eq. 6:

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

F-Measure. This measure takes into account both measures: recall and precision. It can be considered as a weighted average of precision and recall measures with a value ranging between 0 (worst) and 1 (best). F-measure is calculated using Eq. 7:

$$F\text{-}Measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{7}$$

4 Discussion and Results

The performance results of the used classifiers are shown in Table 1 using Recall and Precision metrics that are discussed in the previous section and F-measure that is weighted by the number of classified instances in each class. The Random Forest method achieved the highest accuracy results, with an F-measure of 90.1% and 90.16% of correctly classified instances. In addition, we used the Receiver Operating Characteristic (ROC) metric to evaluate the utilized classifiers. ROC is a curve with points that represent the pair of true positive rate (Sensitivity) and false positive rate (Specificity). A perfect curve is the one that passes through the upper left corner representing 100% sensitivity and 100% specificity. Thus,

Table 1. Performance of classifiers

Classifier name	Accuracy	Precision	Recall	F-measure
J48	87.5012	0.876	0.85	0.873
Random Forest	90.16%	0.905	0.902	0.901
Random Tree	86.8404	0.869	0.868	0.868
SVM	88.2658	0.883	0.883	0.883
Naïve Bayes	82.63	0.846	0.826	0.824
Vote (using Naïve Bayes, Random Forest, SVM)	89.3987	0.899	0.894	0.894
AdaBoost	86.4344	0.867	0.864	0.864
Bagging (using Random Forest)	89.814	0.903	0.898	0.898
IBK	87.2274	0.874	0.872	0.872

the closer the curve is to that corner, the better the accuracy is [44]. As shown in Fig. 2, Random Forest appears to be the best classifier. The high accuracy results demonstrate the efficiency and effectiveness of the extracted features.

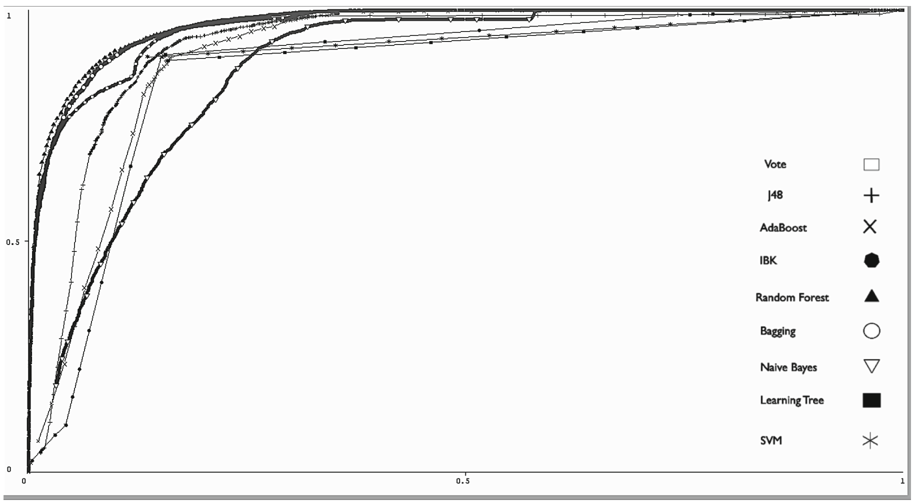


Fig. 2. Performance comparison using ROC

Many studies have utilized the available data from Twitter to build faster influenza surveillance systems [45]. Most of the studies use machine learning methods to distinguish between flu-relevant and irrelevant posts for further analysis. To the best of our knowledge, the classification results we have achieved, using the TF-IDF based classifiers, together with extracted features, show better accuracy compared to the previous work that include tweet classification for

Table 2. Summary of the reviewed flu posts classifiers (Flu-Relevant/Flu-Irrelevant)

Reference	Classifier name	Accuracy	Precision	Recall	F-measure
[32]	SVM and Logistic Regression	N/A	67	87	75.62
[33]					
[35]	Naïve Bayes	N/A	N/A	N/A	83
[34]	SVM	N/A	N/A	N/A	75.6
[36]	SVM	N/A	87.49	92.28	89.68
[37]	Naïve Bayes	70	N/A	N/A	N/A
Proposed Framework	TF-IDF based Random Forest	90.1633	90.5	90.2	90.1

Twitter-based flu surveillance systems. A summary of the performance results of previous works is shown in Table 2. The evaluation of flu tweet classification using the F-measure, which is a weighted average of precision and recall measures, shows that the proposed framework using Random Forest has achieved the highest accuracy value of 90.16%.

5 Conclusion

The analysis of available data from Microblogging sites, such as Twitter, have become very common in event detection and prediction. Recently, many studies have utilized this data to build faster epidemic detection models such as flu outbreak detections. It has been found in our literature survey that most of the models utilize machine learning methods to filter and distinguish between the flu-relevant and irrelevant posts for further analysis. In this study, we have been able to enhance the accuracy of this crucial module for a Twitter-based surveillance system by using TF-IDF for term weighting and including sentiment analysis as part of the used feature vector. The results show that the Random Forest classifier achieved the highest accuracy, with an F-measure of 90.1% and 90.16% of correctly classified instances. Thus, the introduced approach is useful for Twitter-based outbreak detection models/systems.

References

1. Moorhead, S.A., Hazlett, D.E., Harrison, L., Carroll, J.K., Irwin, A., Hoving, C.: A new dimension of health care: systematic review of the uses, benefits, and limitations of social media for health communication. J. Med. Internet Res. **15**(4), e85 (2013)
2. Nurwidyantoro, A., Winarko, E.: Event detection in social media: a survey. In: 2013 International Conference on ICT for Smart Society (ICISS), pp. 1–5. IEEE (2013)
3. Itoh, M., Yokoyama, D., Toyoda, M., Tomita, Y., Kawamura, S., Kitsuregawa, M.: Visual fusion of mega-city big data: an application to traffic and tweets data analysis of metro passengers. In: 2014 IEEE International Conference on Big Data (Big Data), pp. 431–440. IEEE (2014)

4. Wang, X., Zeng, K., Zhao, X.-L., Wang, F.-Y.: Using web data to enhance traffic situation awareness. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 195–199. IEEE (2014)
5. Zhang, S.: Using Twitter to enhance traffic incident awareness. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems, pp. 2941–2946. IEEE (2015)
6. Kosala, R., Adi, E., et al.: Harvesting real time traffic information from Twitter. Procedia Eng. **50**, 1–11 (2012)
7. Abel, F., Hauff, C., Houben, G.-J., Stronkman, R., Tao, K.: Twitcident: fighting fire with information from social web streams. In: Proceedings of the 21st International Conference on World Wide Web, pp. 305–308. ACM (2012)
8. Terpstra, T., de Vries, A., Stronkman, R., Paradies, G.L.: Towards a realtime Twitter analysis during crises for operational crisis management. Simon Fraser University, Burnaby, BC, Canada (2012)
9. Adam, N., Eledath, J., Mehrotra, S., Venkatasubramanian, N.: Social media alert and response to threats to citizens (SMART-C). In: 2012 8th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), pp. 181–189. IEEE (2012)
10. Abel, F., Hauff, C., Houben, G.-J., Stronkman, R., Tao, K.: Semantics + filtering + search = twitcident. Exploring information in social web streams. In: Proceedings of the 23rd ACM Conference on Hypertext and Social Media, pp. 285–294. ACM (2012)
11. 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 (2010)
12. Qusef, A., Ismail, K.: Social media in project communications management. In: 2016 7th International Conference on Computer Science and Information Technology (CSIT), pp. 1–5, July 2016
13. Treboux, J., Cretton, F., Evéquoz, F., Calvé, A.L., Genoud, D.: Mining and visualizing social data to inform marketing decisions. In: 2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA), pp. 66–73, March 2016
14. Wan, S., Paris, C., Georgakopoulos, D.: Social media data aggregation and mining for internet-scale customer relationship management. In: 2015 IEEE International Conference on Information Reuse and Integration (IRI), pp. 39–48, August 2015
15. Burgess, J., Bruns, A.: Twitter archives and the challenges of “big social data” for media and communication research. M/C J. **15**(5) (2012)
16. Yang, B., Guo, W., Chen, B., Yang, G., Zhang, J.: Estimating mobile traffic demand using Twitter. IEEE Wirel. Commun. Lett. **5**(4), 380–383 (2016)
17. Jackoway, A., Samet, H., Sankaranarayanan, J.: Identification of live news events using Twitter. In: Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks, pp. 25–32. ACM (2011)
18. Ishikawa, S., Arakawa, Y., Tagashira, S., Fukuda, A.: Hot topic detection in local areas using Twitter and Wikipedia. In: ARCS Workshops (ARCS), pp. 1–5. IEEE (2012)
19. Petrovic, S., Osborne, M., Lavrenko, V.: The Edinburgh Twitter corpus. In: Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media, pp. 25–26 (2010)
20. Osborne, M., Petrovic, S., McCreadie, R., Macdonald, C., Ounis, I.: Bieber no more: first story detection using Twitter and Wikipedia. In: SIGIR 2012 Workshop on Time-Aware Information Access (2012)

21. Naveed, N., Gottron, T., Kunegis, J., Alhadi, A.C.: Bad news travel fast: a content-based analysis of interestingness on Twitter. In: Proceedings of the 3rd International Web Science Conference, WebSci 2011, pp. 8:1–8:7. ACM, New York (2011)
22. Corley, C.D., Cook, D.J., Mikler, A.R., Singh, K.P.: Text and structural data mining of influenza mentions in web and social media. Int. J. Environ. Res. Public Health **7**(2), 596–615 (2010)
23. Hwang, M.-H., Wang, S., Cao, G., Padmanabhan, A., Zhang, Z.: Spatiotemporal transformation of social media geostreams: a case study of Twitter for flu risk analysis. In: Proceedings of the 4th ACM SIGSPATIAL International Workshop on GeoStreaming, pp. 12–21. ACM (2013)
24. Polgreen, P.M., Chen, Y., Pennock, D.M., Nelson, F.D., Weinstein, R.A.: Using internet searches for influenza surveillance. Clin. Infect. Dis. **47**(11), 1443–1448 (2008)
25. Goel, S., Hofman, J.M., Lahaie, S., Pennock, D.M., Watts, D.J.: Predicting consumer behavior with web search. Proc. Natl. Acad. Sci. **107**(41), 17486–17490 (2010)
26. Scharkow, M., Vogelgesang, J.: Measuring the public agenda using search engine queries. Int. J. Public Opin. Res. **23**(1), 104–113 (2011)
27. Dugas, A.F., Hsieh, Y.-H., Levin, S.R., Pines, J.M., Mareiniss, D.P., Mohareb, A., Gaydos, C.A., Perl, T.M., Rothman, R.E.: Google flu trends: correlation with emergency department influenza rates and crowding metrics. Clin. Infect. Dis. **54**(4), 463–469 (2012)
28. Morrison, J.L., Breitling, R., Higham, D.J., Gilbert, D.R.: GeneRank: using search engine technology for the analysis of microarray experiments. BMC Bioinform. **6**(1), 1 (2005)
29. Ginsberg, J., Mohebbi, M.H., Patel, R.S., Brammer, L., Smolinski, M.S., Brilliant, L.: Detecting influenza epidemics using search engine query data. Nature **457**(7232), 1012–1014 (2009)
30. Lee, K., Agrawal, A., Choudhary, A.: Real-time disease surveillance using Twitter data: demonstration on flu and cancer. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1474–1477. ACM (2013)
31. Suh, B., Hong, L., Pirolli, P., Chi, Ed 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 (2010)
32. Broniatowski, D.A., Paul, M.J., Dredze, M.: National and local influenza surveillance through Twitter: an analysis of the 2012–2013 influenza epidemic. PLoS One **8**(12), e83672 (2013)
33. Lamb, A., Paul, M.J., Dredze, M.: Separating fact from fear: tracking flu infections on Twitter. In: North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013), June 2013
34. Aramaki, E., Maskawa, S., Morita, M.: Twitter catches the flu: detecting influenza epidemics using Twitter. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1568–1576. Association for Computational Linguistics (2011)
35. Santos, J.C., Matos, S.: Analysing Twitter and web queries for flu trend prediction. Theor. Biol. Med. Model. **11**(1), S6 (2014)
36. Cui, X., Yang, N., Wang, Z., Cheng, H., Zhu, W., Li, H., Ji, Y., Liu, C.: Chinese social media analysis for disease surveillance. Pers. Ubiquit. Comput. **19**(7), 1125–1132 (2015)

37. Byrd, K., Mansurov, A., Baysal, O.: Mining Twitter data for influenza detection and surveillance. In: IEEE/ACM International Workshop on Software Engineering in Healthcare Systems (SEHS), pp. 43–49. IEEE (2016)
38. Sanders, N.J.: Sanders-Twitter Sentiment Corpus (2011). http://www.sananalytics.com/lab/twitter-sentiment/. Accessed 20 Oct 2017
39. Amazon MTurk: Amazon Mechanical Turk (MTurk)
40. Bird, S.: NLTK: the natural language toolkit. In: Proceedings of the COLING/ACL on Interactive Presentation Sessions, pp. 69–72. Association for Computational Linguistics (2006)
41. Singh, J., Gupta, V.: A systematic review of text stemming techniques. Artif. Intell. Rev. **48**(2), 157–217 (2017)
42. Joachims, T.: A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. Technical report, Carnegie-mellon univ pittsburgh pa dept of computer science (1996)
43. Loria, S., Keen, P., Honnibal, M., Yankovsky, R., Karesh, D., Dempsey, E., et al.: TextBlob: simplified text processing. Secondary TextBlob: Simplified Text Processing (2014)
44. Zweig, M.H., Campbell, G.: Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. Clin. Chem. **39**(4), 561–577 (1993)
45. Alessa, A., Faezipour, M.: A review of influenza detection and prediction through social networking sites. Theor. Biol. Med. Model. **15**(2), 1–27 (2018)

Adaptive Adjacency Kanerva Coding for Memory-Constrained Reinforcement Learning

Wei Li(✉) and Waleed Meleis

Northeastern University, Boston, MA 02115, USA
li.wei@husky.neu.edu, meleis@ece.neu.edu

Abstract. When encountering continuous, or very large domains, using a compact representation of the state space is preferable for practical reinforcement learning (RL). This approach can reduce the size of the state space and enable generalization by relating similar or neighboring states. However, many state abstraction techniques cannot achieve satisfactory approximation quality in the presence of limited memory resources, while expert state space shaping can be costly and usually does not scale well. We have investigated the principle of Sparse Distributed Memories (SDMs) and applied it as a function approximator to learn good policies for RL. This paper describes a new approach, adaptive adjacency in SDMs, that is capable of representing very large continuous state spaces with a very small collection of prototype states. This algorithm enhances an SDMs architecture to allow on-line, dynamically-adjusting generalization to assigned memory resources to provide high-quality approximation. The memory size and memory allocation no longer need to be manually assigned before and during RL. Based on our results, this approach performs well both in terms of approximation quality and memory usage. The superior performance of this approach over existing SDMs and tile coding (CMACs) is demonstrated through a comprehensive simulation study in two classic domains, Mountain Car with 2 dimensions and Hunter-Prey with 5 dimensions. Our empirical evaluations demonstrate that the adaptive adjacency approach can be used to efficiently approximate value functions with limited memories, and that the approach scales well across tested domains with continuous, large-scale state spaces.

Keywords: Reinforcement learning · Function approximation
Kanerva coding · Dynamic generalization

1 Introduction

Reinforcement learning (RL) techniques have been shown to develop effective policies for diverse tasks and application domains. However, in general, applying RL to practical problems is challenging in the presence of continuous, large-scale

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 187–201, 2018.
https://doi.org/10.1007/978-3-319-96136-1_16

state spaces. In this case, the size of the state space can grow exponentially with the number of state variables, causing a proportional increase in the size of the value table needed to store the value functions. The training time needed to explore large state spaces is expensive and should be reduced. In the domains with continuous state spaces, it is impractical to enumerate and maintain all possible states. Associating approximating techniques with RL algorithms has enabled the approach to be effectively applied to many practical problems. However, some challenges still remain.

In the literature, a number of works [5,14,17] have proposed applying approximating techniques to large state-action spaces. Function approximation is a landmark technique that can effectively reduce the size of the search space by representing it with a compact and parameterized version [6]. The value table that stores value functions is replaced by an approximate and compact table. Tile coding (CMACs) [18], adaptive tile coding [12,19] and tree-based state space discretization [2] are prominent function approximation approaches that have been evaluated in various problem domains. However, when solving practical real-world problems, limitations on the coding schemes used in those approaches, e.g., inflexible partitions or computationally costly abstractions of the state space, make them hard to scale well or guarantee high performance in domains that are very large, or have continuous state and action spaces.

Kanerva coding [8], also known as Sparse Distributed Memories (SDMs), has emerged as an efficient and viable solution to deal with complex, large state spaces. This technique considers such a setting that the whole state space is represented by a substantially small subset of the state space. Since a set of states is used as the basis for approximation, the complexity of this scheme is dependent only on the number of states in the subset and not on the number of dimensions of the state space, making it suitable for solving problems with continuous, large state spaces. Kanerva-based learners' effectiveness in problems with continuous, large state spaces has been evaluated by [11,20]. However, to improve the performance of Kanerva coding algorithms, the allocation of the states in the subset need to be optimally arranged based on considered visited state areas which is still an ongoing research field and needs further exploration.

In this paper, based on the principles of SDMs, we describe a function approximator that adaptively adjusts the radii of receptive fields of selected *memory locations* (also called *prototypes*). Such adjustments can directly change the generalization abilities of memory locations to cover certain visited states when necessary, making it possible to fully utilize limited memory resources while capturing enough of the complexity of visited states. This approach can learn a good policy through its attempt to produce a high-quality generalization of the selected memory locations by appropriately adjusting and utilizing various levels of generalization for each memory location. Experimental results show the superior performance of this proposed approach both in terms of memory usage and approximation quality.

The rest of the paper is organized as follows. In Sect. 2, we give an overview of the RL Sarsa algorithm [18] and present the principals and limitations of existing Kanerva-based approaches that serve as a linear function approximator when applying RL algorithms in problem domains with continuous, very large state spaces. In Sect. 3, we introduce our adaptive adjacency approach for flexible memory adjacency adjustments in SDMs, and then we describe the implementation of this approach. In Sect. 4, we show our experimental results and analysis. Finally, we give our conclusions in Sect. 5.

2 Related Work

RL has been successfully applied to a wide range of sequential decision problems and problem domains, such as video streaming [3], computer game playing [7], physical science [10] and networking [11,13]. RL can learn on-line without the need for initial input data due to its ability to learn through a trial-and-error mechanism. In RL, the learning agent receives a reward after performing particular actions in a state. It transits to a new state based on the environment's response to currently performed action. Its goal is to develop a policy, a mapping from a state space to an action space, that maximizes the long-term rewards. Many RL techniques have been proposed, i.e., Sarsa and Q-learning.

2.1 Sarsa Algorithm

Sarsa [18] is an on-policy temporal-difference (TD) control algorithm in RL. At each time step t, the learning agent observes a state s_t from the environment and receives a reward r_{t+1} after applying an action a_t to this state s_t. Current state s_t then transitions to next state s_{t+1} and the learning agent chooses an action a_{t+1} at state s_{t+1}. Sarsa algorithm updates the expected discounted reward of each state-action pair, in this case, the action-value function $Q(s_t, a_t)$, as follows:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha_t(s_t, a_t)[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)], \tag{1}$$

where $\alpha_t(s_t, a_t)$ is the learning rate ($0 \leq \alpha_t(s_t, a_t) \leq 1$), and γ is the discount factor ($0 < \gamma \leq 1$). The value of each action-value function $Q(s_t, a_t)$ is learned and stored in a table, called Q-table. The use of Q-table is suitable for small state-action space but impractical with large or continuous state-action spaces due to the considerable memory consumption and unbearable long training time.

Although Sarsa performs well in terms of both learning quality and convergence time when solving problems with small state and action spaces, Sarsa performs poorly when applied to large-scale, continuous state and action spaces due to its inefficient use of Q-table. Function approximation techniques are therefore proposed to solve this problem where, on the one hand, the values of a relatively much small set of states/prototypes are stored instead of storing the Q-table for all visited states, and on the other hand, the knowledge among similar states are generalized and transferable.

2.2 Kanerva-Based Learning

Kanerva Coding (SDMs). The idea of Kanerva coding [18] considers to use a relatively small set of states as *prototypes* to store and estimate the value functions. In Kanerva coding, a state s or prototype p_i consists of n state-variables and each state-variable describes a measurement in one dimension with a numeric value. A state s and a prototype p_i are said to be *adjacent* if s and p_i differ in at most a given number of state-variables, for example differing in at most one state-variable. The *membership grade* $\mu(s, p_i)$ of state s with respect to prototype p_i is defined where $\mu(s, p_i)$ is equal to 1 if s is adjacent to p_i, and 0 otherwise. Kanerva coding maintains a set of k prototypes as parameterized elements for approximation and a value $\theta(p_i, a)$ is stored and updated for each prototype p_i with respect to the action a.

The approximation of the value of a state-action pair (s, a) is calculated by a linear combination of θ-values of all adjacent prototypes of state s, defined as follows:

$$\hat{Q}(s,a) = \sum_{p_i \in D_\mathrm{s}} \theta(p_i, a)\mu(s, p_i), \tag{2}$$

where D_s is the set of adjacent prototypes with respect to state s. The θ-value of each prototype p_i with respect to action a is updated by the rule of Sarsa algorithm as follows:

$$\theta(p_i, a) \leftarrow \theta(p_i, a) + \frac{\mu(s, p_i)}{\sum_{p_k \in D_s} \mu(s, p_k)} \alpha(s,a)[r + \gamma \hat{Q}(s', a') - \hat{Q}(s,a)]. \tag{3}$$

Dynamic Kanerva Coding. Kanerva coding works well in large and complex problem domains. However its performance is sensitive to the allocation of prototypes in use [9]. The set of prototypes should be appropriately selected from the state space, i.e., choosing ones that are well distributed around the trajectories of visited states, otherwise, the quality of function approximation could be greatly degraded. In fact, selecting/reallocating the set of prototypes before and/or during the learning process to maintain a sufficient complexity of function approximation is the central issue that needs to be solved for any Kanerva-based algorithms.

There are two classes of solutions typically used to solve this problem. The first class of approaches starts with an empty set of prototypes and then incrementally builds up the set that covers areas of interest in the state space by adding a certain number of prototypes. One heuristic used to add prototypes is to add a new prototype if it is at least a certain distance away from all existing prototypes [4,16]. The heuristic is inflexible. When the complexity of the required approximations is already satisfied, there will be no need to introduce additional prototypes. This heuristic could introduce many redundant prototypes resulting in slow learning and extra memory costs. Another approach is to add a collection of new prototypes from the neighborhood area of a sample of recently visited states, provided each of them is far enough from all existing prototypes [15]. However, this heuristic requires a sophisticated strategy to select appropriate prototypes from the neighborhood area of a sample of visited states, particularly when avoiding conflicts with existing prototypes. In addition, both heuristics incur considerable computational cost, i.e., calculating and leveraging the distance from the visited state to all existing prototypes. The second class of approaches [1,21] starts with an initial set of randomly selected prototypes. It has much less computational cost. To guarantee efficiency in function approximations, it tends to replace poorly-performing/rarely utilized prototypes with promising prototypes to gradually adjust the allocation of the original set of prototypes. However, prototypes should not be replaced only based on their periodically-observed performance and too-frequent prototype replacements could lose previously-learned information. In real implementation, both classes of approaches may combine together to pursue even better performance.

Our adaptive adjacency approach addresses the weaknesses of both classes of techniques while achieving satisfactory performance across multiple problem domains. This approach also lowers computational cost by eliminating the operations to add and delete prototypes.

3 Adaptive Adjacency Kanerva Coding

3.1 Prototype Adjacencies

In our experiments, the state space is continuous and each input sample data or prototype is represented by a sequence of real values, i.e., each dimension is represented by a real value. The distance between an input sample data $s = \langle v_1, ..., v_n \rangle$ and one of the prototypes, $p_i = \langle h_1, ..., h_n \rangle$, in the prototype set is defined by the following distance functions:

$$d(s, p_i) = \sum_{j=1}^{n} d_j(s, p_i), \tag{4}$$

$$d_j(s, p_i) = \begin{cases} 0 : |v_j - h_j| \leq \sigma_i \beta_j, \\ 1 : \text{otherwise.} \end{cases} \tag{5}$$

Equation (5) is very important in our approach. In (5), β_j is the basis radius of the receptive field in j_{th} dimension, and σ_i is the *adjacency factor* that is multiplied by β_j to adjust the radii of the receptive field of each prototype p_i. Note that $d(s, p_i)$ represents the number of state variables at which state s and prototype p_i differ by more than $\sigma_i \beta_j$. The value of σ_i can be adaptively changed by our algorithm for each prototype p_i during the regular learning process. In our algorithm, β_j does not change.

We define the *membership grade* $\mu(s, p_i)$ of state s with respect to prototype p_i to be a continuous function of the distance between s and p_i as follows:

$$\mu(s, p_i) = e^{-d(s, p_i)}. \tag{6}$$

Instead of a binary membership grade, we use a continuous membership grade in $[0, 1]$ to assign a weighted value update (based on distance) to the θ-value of

Algorithm 1. Adaptive Adjacency Kanerva Coding with Sarsa Algorithm

```
Input: $\boldsymbol{p}$: a set of randomly selected prototype samples, $\boldsymbol{\theta}$: associated with $\boldsymbol{p}$
       and initialized to $\mathbf{0}$, $\boldsymbol{\sigma}$: associated with $\boldsymbol{p}$ and initialized to $\mathbf{1}$, $\boldsymbol{\beta}$: $\beta_j$ is a
       basis radius of receptive field in $j_{th}$ dimension of the state space.
Output: learned $\boldsymbol{\theta}$-values and adjusted $\boldsymbol{\sigma}$-values
1  Procedure Main()
2      for each episode do
3          Initialize $s$
4          Choose $a$ in $s$ based on policy derived from $\boldsymbol{\theta}$
5          for each step of episode do
6              Take action $a$, observe $r$, $s'$
7              $\boldsymbol{p}_{\text{activated}}$ = the set of prototypes activated by $s'$
8              $M$ = the size of $\boldsymbol{p}_{\text{activated}}$
9              if $M \geq N$ then
10                 Choose $a'$ in $s'$ based on $\boldsymbol{p}_{\text{activated}}$ and $\epsilon$-greedy approach
11                 Update $\theta$ values with (3)
12             else
13                 $\boldsymbol{p}_{\text{activated}}$ = AdaptiveAdjacency($\boldsymbol{\sigma}$, $\boldsymbol{p}$, $s'$, $\boldsymbol{p}_{\text{activated}}$)
14                 Choose $a'$ in $s'$ based on $\boldsymbol{p}_{\text{activated}}$ and $\epsilon$-greedy approach
15                 Update $\theta$ values with (3)
16             Set $s = s'$ and $a = a'$
17         Until $s$ is terminal
18 Procedure AdaptiveAdjacency($\boldsymbol{\sigma}$, $\boldsymbol{p}$, $s'$, $\boldsymbol{p}_{\text{activated}}$)
19     Initialize the factor $k$ to 0
20     for each round of adjusting radii of receptive field do
21         Increase $k$ by 1
22         for each prototype $p_i$ in $\boldsymbol{p}$ do
23             if $p_i$ is not in $\boldsymbol{p}_{\text{activated}}$ and $\sigma_i$ has not been modified more than $L$
               times then
24                 Increase $\sigma_i$ by a multiplicative factor, $(1 + k\phi)$, where $\phi \in$[0,1]
25                 if $p_i$ is activated by its newly increased radii of receptive field
                   then
26                     Add $p_i$ to $\boldsymbol{p}_{\text{activated}}$
27                     $M$ = the size of $\boldsymbol{p}_{\text{activated}}$
28                     if $M \geq N$ then
29                         Return
30                 else
31                     Reset $\sigma_i$ to its previously not-yet-increased value
```

each adjacent prototype. Note that the membership grade of a state with respect to an identical or 0-distance prototype would be 1, and the membership grade of a state with respect to a distant prototype approaches to 0. And a state s and a prototype p_i are said to be *adjacent* if $d(s, p_i) \leq \left\lfloor \frac{n}{2} \right\rfloor$. We also say that a prototype is *activated* by a state if the prototype is adjacent to the state.

The estimates of the Q-values and the updates to the θ-values use the same definitions described in (2) and (3), respectively. In (5), parameter σ_i plays an important role in adjusting the radius of the receptive field in each dimension, giving each prototype the ability to adjust its sensitivity to visited states. The ability of each prototype to generalize can now be adjusted, based on information gathered during the learning process.

3.2 Adaptively Changing Adjacencies

Our approach begins with a set of randomly-selected prototypes from the state space. Each prototype p_i has a corresponding factor σ_i whose value allows for on-line adjustments that directly change the radii of the receptive field of prototype p_i. A parameter N (denoted as *active parameter*), the minimum number of adjacent prototypes for a visited state, is used to control the amount of generalization that is provided.

For each encountered state, the algorithm adjusts prototypes' receptive fields so that state is adjacent to at least N prototypes. More specifically, when encountering a state s, if N or more prototypes are adjacent to s, we perform value updates using the regular SDMs routine. However, if only M prototypes, where $0 \leq M < N$, are adjacent to s, our approach enlarges the receptive fields of certain nearby prototypes enough to allow these prototypes to be adjacent to s. This allows s to meet the adjacency requirement.

A detailed description of our adaptive adjacency approach is presented in Algorithm 1. Starting from line 20, in each round of increments, the algorithm provisionally adjusts the boundary of the receptive field for each prototype p_i that is not activated by the current state by increasing σ_i by a multiplicative factor that is >1. This process continues through additional rounds of increments until the state is adjacent to at least N prototypes. The algorithm then fixes the receptive field boundaries for all newly activated prototypes, and resets the receptive field boundaries for the remaining prototypes that failed to be activated back to the values held before the increments began. In each round, the multiplicative factor used to augment the receptive fields increases.

Line 23 of Algorithm 1 implements a constraint that ensures that the radii of receptive field of a particular prototype can only be adjusted a limited number of times, i.e., at most L times. This rule guarantees a fine-grained generalization for the set of prototypes and can avoid over-generalization.

4 Experimental Evaluation

We first present an experimental evaluation of our adaptive adjacency Kanerva coding (AdjacencyK) algorithm with Sarsa on the Mountain Car domain [18], a classic RL benchmark. The problem has a continuous state space with 2 state variables: the car position and velocity. We then present an experimental evaluation using a more complex benchmark, a variant of the Hunter-Prey domain that has a continuous, much large state space with 5 state variables (see details in Sect. 4.2). We compare the performance of AdjacencyK algorithm with that of popular CMACs algorithm, pure Kanerva coding (also called traditional SDMs, denoted as PureK) as well as one commonly used dynamic Kanerva coding algorithm (an approach that selectively deletes and generates certain expected prototypes proposed and utilized in [1,21], denoted as DynamicK).

4.1 Evaluate Performance with Mountain Car Domain

In a Mountain Car task, the learning agent attempts to learn a policy to drive an underpowered car up a steep hill. Each task consists of a sequence of episodes, and each episode ends and resets to the start state if the goal is achieved, i.e., the car reaches the top position of the hill, or if the agent-environment interaction exceeds the maximal number of time steps allowed in an episode.

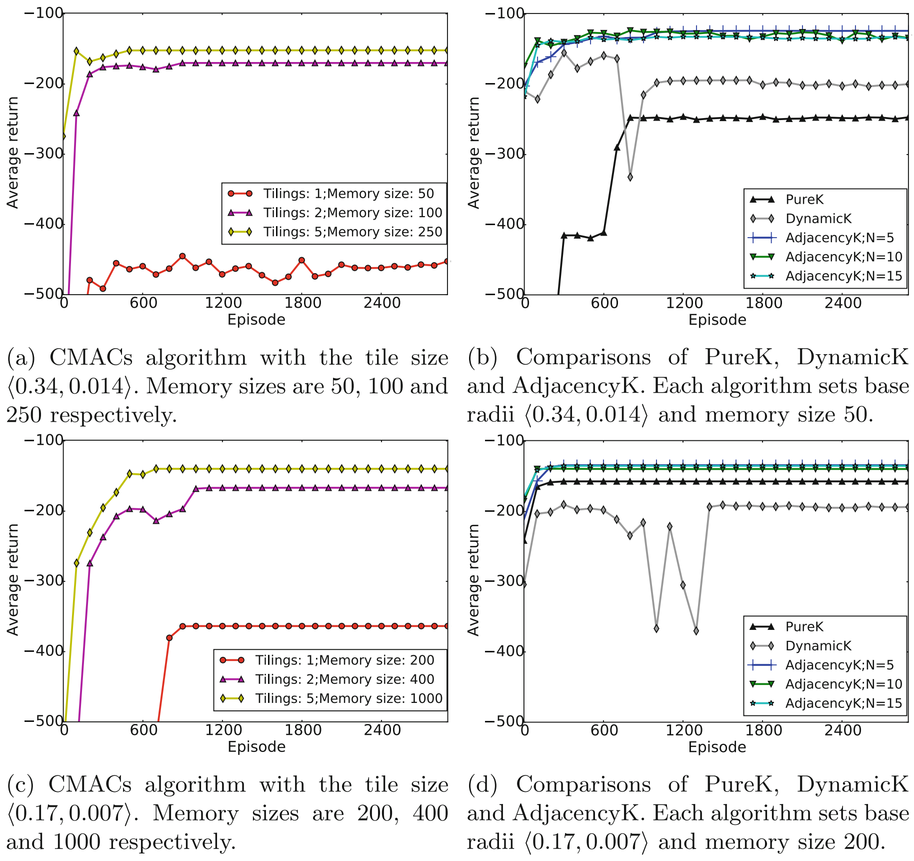
Tile coding (or CMACs) has been successful in helping RL algorithm learn policies for the Mountain Car domain [18]. The performance of CMACs relies largely on obtaining an effective memory layout on the state space. In CMACs, a *tiling* exhaustively partitions the whole state space into *tiles* and an input state sample falls within the receptive field of exactly one tile of a tiling. The receptive field of a tile indicates its generalization ability. To avoid over-generalization and to increase the resolution of needed approximations, typically a collection of overlapped and slightly offset tilings is used.

We ran experiments of our proposed AdjacencyK algorithm on the standard Mountain Car domain and then analyzed its performance over CMACs, PureK and DynamicK algorithms. All our experiments start with the same initial state in which the car is at rest at the bottom of the valley and its velocity is 0. Each prototype's receptive field radius in each dimension in AdjacencyK, PureK and DynamicK is set equal to the size of a tile in corresponding dimension in CMACs. Since the number of tilings in CMACs greatly affects the agent's ability to distinguish states in the state space, our experiments test different number of tilings to present a complete set of empirical results. We also explore the differences in performance when using various values of parameter N in proposed AdjacencyK algorithm.

The PureK and DynamicK have the same initial set of randomly selected prototypes as the one used in our AdjacencyK approach and the initial radii of the receptive fields are also the same. Note that the radii in PureK and DynamicK are unable to change during learning.

The results of applying CMACs, PureK, DynamicK and AdjacencyK algorithms to the Mountain Car domain are shown in Fig. 1. The results are averaged on 8 repeated runs and each episode starts with the same initial state. The best selected RL parameters are set as: $\alpha = 0.5$, $\epsilon = 0.0$, and $\gamma = 1.0$. We set ϕ to 0.5 and L to 1 in AdjacencyK algorithm. The results show that the AdjacencyK approach outperforms CMACs, PureK and DynamicK both in terms of learning quality and memory usage. As demonstrated by Fig. 1a and b, the adaptive adjacency approach learns the task faster and converges to a higher return in a shorter time than CMACs, PureK and DynamicK algorithms.

We also evaluate the sensitivity of the performance of considered algorithms to variations in the settings of important parameters. In Fig. 1b, we show that different values of active parameter N, i.e., 5, 10, 15, in proposed AdjacencyK



(a) CMACs algorithm with the tile size $\langle 0.34, 0.014\rangle$. Memory sizes are 50, 100 and 250 respectively.

(b) Comparisons of PureK, DynamicK and AdjacencyK. Each algorithm sets base radii $\langle 0.34, 0.014\rangle$ and memory size 50.

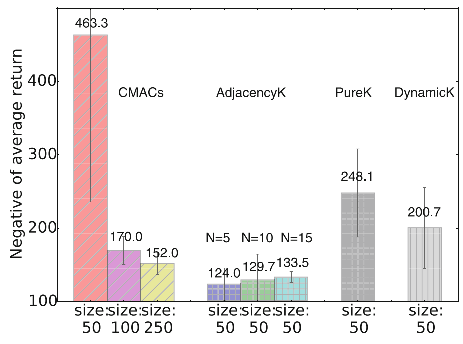
(c) CMACs algorithm with the tile size $\langle 0.17, 0.007\rangle$. Memory sizes are 200, 400 and 1000 respectively.

(d) Comparisons of PureK, DynamicK and AdjacencyK. Each algorithm sets base radii $\langle 0.17, 0.007\rangle$ and memory size 200.

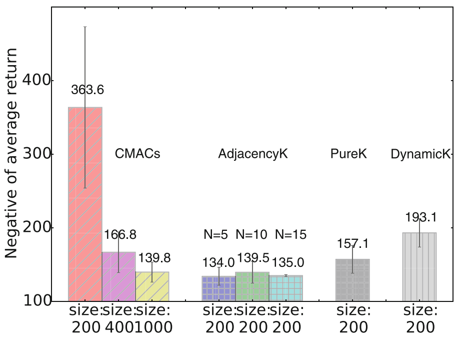
Fig. 1. Comparisons of average returns of 4 algorithms with different sizes of tiles and radii of receptive fields in the Mountain Car domain.

algorithm have only a small impact on average returns and therefore do not need carefully tuned. Furthermore, the two graphs in the top row of Fig. 1 show data assuming the same size base receptive field (or tile size), and the two graphs in the bottom row show data assuming a smaller size base receptive field (or tile size). The results in Fig. 1c and d demonstrate that the adaptive adjacency approach still outperforms CMACs, PureK and DynamicK algorithms both in terms of learned policies and memory usage as the sizes of base radii of receptive fields (or tiles) decrease from $\langle 0.34, 0.014\rangle$ to $\langle 0.17, 0.007\rangle$. Interestingly, based on the new settings on the radii and memory size, Fig. 1d shows that the performance of PureK was largely improved, e.g., PureK with memory size of 200 learned a slightly better policy than CMACs with memory size of 400.

Figure 2 demonstrates the memory savings of our proposed adaptive adjacency approach (AdjacencyK) with two different settings of the sizes of base radii of receptive fields. With each setting, we show the negative of average returns of



(a) Base radii ⟨0.34, 0.014⟩



(b) Base radii ⟨0.17, 0.007⟩

Fig. 2. Memory savings of AdjacencyK with two different sizes of base radii.

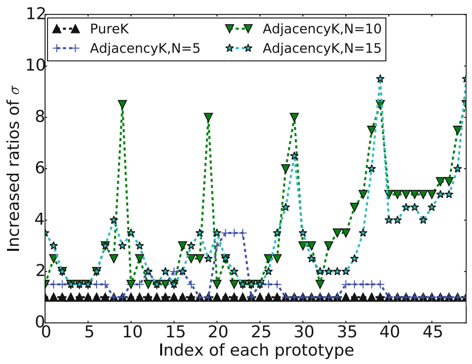


Fig. 3. Changes of the receptive field of each prototype with PureK and AdjacencyK in the Mountain Car task. The base radii are ⟨0.34, 0.014⟩. The allocations of prototypes are exactly same for both algorithms when the learning starts.

CMACs with various memory sizes (refer to the number of tiles), AdjacencyK with a fixed memory size (refer to the number of prototypes) but various values of N, and PureK as well as DynamicK with the same fixed memory size as AdjacencyK after 1500 episodes of learning in our experiments. When testing with one setting of base radii of receptive fields (or tiles), i.e., ⟨0.34, 0.014⟩, the left three bars in Fig. 2a show that the performance of CMACs is poor and unstable when the memory size is small (i.e., 50), and its performance improves significantly when the memory size is relatively large (i.e. 250) which is reasonable. And as demonstrated by the three clustered bars of AdjacencyK algorithm, our adaptive adjacency approach can achieve even better performance than CMACs when only using 20% of the memories. At the same time, it can be observed in Fig. 2a that given the same initial set of prototypes, our approach outperforms both Kanerva-based algorithms, PureK and DynamicK, with the same memory usages. And generally, these observations are also true for results shown in Fig. 2b that has a different setting of base radii of receptive fields (or tiles).

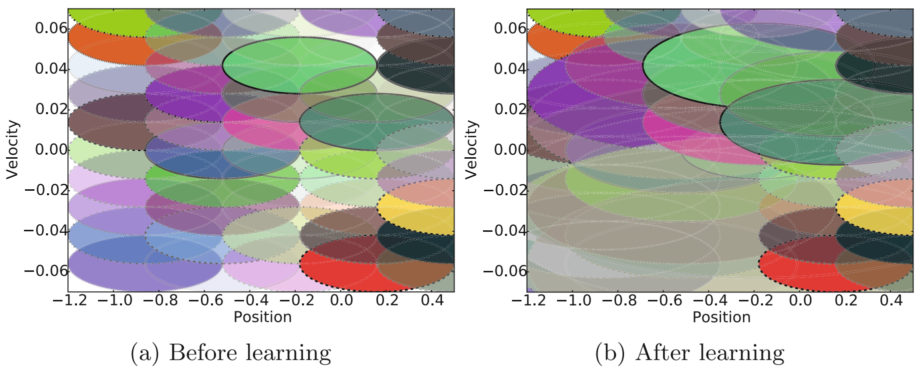


Fig. 4. Changes to the coverages of receptive fields for all prototypes with the AdjacencyK approach in a Mountain Car task.

We argue that the improved performance in our proposed approach results from changes in the sizes of receptive fields of prototypes during learning. Figure 3 shows these changes to the sizes of receptive fields across all prototypes. As shown in Fig. 3, initialized with the same set of prototypes and base radii, PureK algorithm does not change the sizes of receptive fields during RL while our proposed algorithm allows the receptive fields appropriately changed (see the rises of the curves in the graph) and thus all prototypes' generalization capabilities are flexibly adjusted so that required approximation complexity is guaranteed for input state samples. In other words, during learning, each input state would be adjacent to a sufficient number of prototypes and each prototype has the chance to be activated if being adjusted to have a required size of receptive field. Note that a larger value of N means that input states expect to have higher complexity of approximations, i.e., have more adjacent prototypes for function predictions. Therefore, a larger N implies that the receptive fields of prototypes will need to be increased more so that their generalization abilities are enhanced in order to be more easily activated by input states. As shown in Fig. 3, the learning agent with a bigger N value, i.e., $N = 10$ or $N = 15$, increases its prototypes' radii of receptive fields more obviously than that with $N = 5$.

In our experiments, the adaptive adjacency approach gave similar good learning results when setting N to 5, 10 or 15. We argue that these good results are a result of having a collection of prototypes with necessarily adjusted sizes of receptive fields that overlap with visited input states sufficiently. Figure 4 shows the changes to the sizes of receptive fields of 50 uniformly distributed prototypes when performing the adaptive adjacency approach to a Mountain Car task. Note that all the data depicting the radii changes in this figure came from one of our experiments that learned a good policy. The detailed results of this experiment are already shown in Fig. 1b and Fig. 3 in which the base radii are $\langle 0.34, 0.014 \rangle$, the memory size is 50 and the active parameter N is 5.

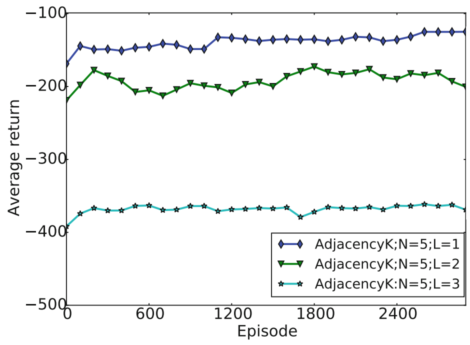


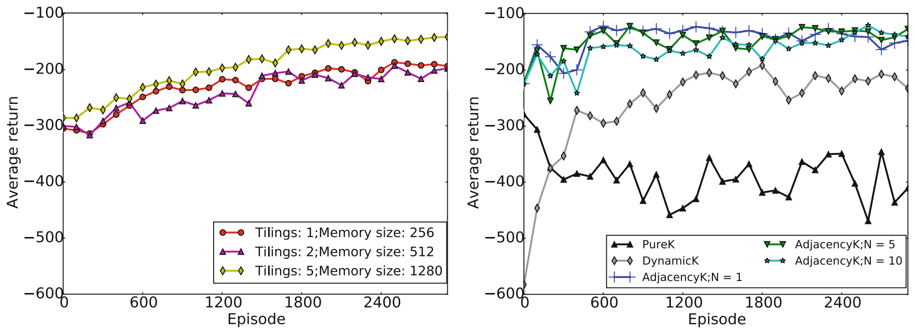
Fig. 5. Average returns of adaptive adjacency approach with different settings of L. Base radii for each prototype is $\langle 0.34, 0.014\rangle$, and memory size is 50.

The constraint L defined in our algorithm that limits the number of changes to the adjacency factor σ can have a large impact on the learning performance. Figure 5 shows the results of applying our algorithm with 3 different L values to a Mountain Car task. It shows that our algorithm achieves better learning results when using a smaller L. The reason is that a smaller L allows certain distant prototypes to have opportunities to change their receptive fields and thus provides a collection of prototypes with receptive fields of much varying sizes that give finer-grained generalization on the state space. Therefore, in order to achieve best learning quality, we set L to 1 in all our experiments.

4.2 Evaluate Performance with Hunter-Prey Domain

We also evaluate the performance of proposed adaptive adjacency approach on a variant of Hunter-Prey task introduced in [15]. In our experiments, we use one prey and two hunters where the hunters work cooperatively to capture the prey and the prey needs to learn a policy to avoid being captured, e.g., the prey can kill a hunter if only this alone hunter is close enough to the prey. The state space in this task consists of 4 continuous state variables in which the position of each hunter is described by 2 continuous variables, i.e., a radial coordinate and an angular coordinate in a polar coordinate system with the prey as the reference point, and one extra integer variable indicating the number of alive hunters.

To capture the prey, both hunters should approach close enough to the prey, i.e., within 5 units, and the two angles between both hunters need to be $\leq \pi + 0.6$ radians. Each hunter's movements follow a predefined stochastic strategy similar to a ϵ-greedy approach. For example, in the radial direction, each hunter moves 5 units towards the prey if this movement would not get it killed (otherwise, it moves 5 units away from the prey), during all but some fraction ϵ of the time when each hunter makes a 0.2 radians' movement clockwise or counterclockwise. We set ϵ to 0.7 in our experiments to make the hunters less greedy to approach to the prey, otherwise the prey is very likely to kill an individual hunter even



(a) CMACs algorithm with the tile size $\langle 125, 1.57\rangle^2$. Memory sizes are 256, 512 and 1280 respectively.

(b) Comparisons of PureK, DynamicK and AdjacencyK. Each algorithm sets base radii $\langle 125, 1.57\rangle^2$ and memory size 256.

Fig. 6. Comparisons of average returns of 4 algorithms in the Hunter-Prey domain.

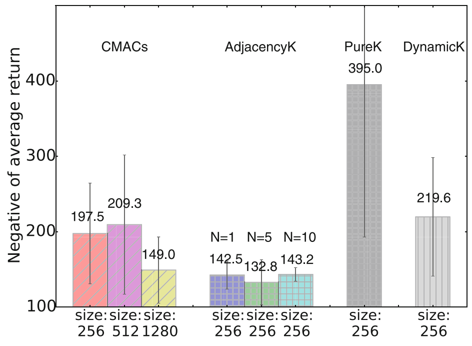


Fig. 7. Memory savings of the AdjacencyK algorithm

with random movements, giving an unexpected high return. The prey moves 5 units per time step in 4 available directions: up, right, down, left in the polar coordinate system. The episode starts with an initial state in which each hunter is randomly placed on a cycle of radius 500 units centered on the prey. If the prey kills one hunter (reward of 1) or gets captured (reward of -200), the episode ends, otherwise, the interaction between prey and hunters continues (reward of -1 per time step).

We show experimental results of our AdjacencyK algorithm and compare its performance with CMACs, PureK and DynamicK algorithms in Figs. 6 and 7. The experimental configuration is similar to that of the Mountain Car task and RL parameters are also optimally selected. The ϕ is set to 0.3 and L is set to 1 in AdjacencyK algorithm. The AdjacencyK approach obviously learns better policies than CMACs, PureK and DynamicK algorithms. PureK algorithm has the worst average returns. It cannot learn the task well with a memory size of

256. Although DynamicK has a better performance than PureK, it still cannot beat CMACs. The CMACs algorithm learns slower than AdjacencyK and if their memory sizes are both 256, the learned average return of AdjacencyK ($N = 1$) is 32.8% better than that of CMACs. Note that the CMACs algorithm needs 5-times of the memories, i.e., 1280, to obtain a comparable performance with our algorithm. In other words, the memory usage of our algorithm is 80% smaller than that of CMACs while maintaining a similar performance.

5 Conclusion

In this paper, we extended the architecture of SDMs by using the adaptive adjacency approach as a practical function approximator, and aimed to use RL with this new function approximator to solve complex tasks with continuous, large state spaces. Our new approach enables the generalization capabilities of prototypes to be dynamically adjusted to provide needed complexities of approximations for all input states. This approach supports flexible shaping of the receptive fields of those preselected memory locations in order to cover the areas of interest in the state space. On the one hand, limited memory resources are more efficiently utilized to accomplish qualified value functions' storage and retrieval. On the other hand, performance is less sensitive to the sizes of the receptive fields as well as memory allocations and reallocations.

Our experimental studies demonstrate that our adaptive adjacency approach learns better policies in RL than the popular CMAC algorithm and other existing algorithms in SDMs over two benchmarks. Moreover, this approach has a concise mechanism that is easy to implement, is flexible and scalable in various domains, and does not require expert tunings or prior knowledge about the state space. Finally, the approach works very well when memory resources are constrained.

References

1. Allen, M., Fritzsche, P.: Reinforcement learning with adaptive Kanerva coding for Xpilot game AI. In: 2011 IEEE Congress of Evolutionary Computation (CEC), pp. 1521–1528. IEEE (2011). https://doi.org/10.1109/CEC.2011.5949796
2. Chernova, S., Veloso, M.: Tree-based policy learning in continuous domains through teaching by demonstration. In: Proceedings of Workshop on Modeling Others from Observations (MOO 2006) (2006)
3. Chiariotti, F., D'Aronco, S., Toni, L., Frossard, P.: Online learning adaptation strategy for dash clients. In: Proceedings of the 7th International Conference on Multimedia Systems, p. 8. ACM (2016). https://doi.org/10.1145/2910017.2910603
4. Forbes, J.R.N.: Reinforcement Learning for Autonomous Vehicles. University of California, Berkeley (2002)
5. Frommberger, L.: Qualitative Spatial Abstraction in Reinforcement Learning. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16590-0
6. Geist, M., Pietquin, O.: Algorithmic survey of parametric value function approximation. IEEE Trans. Neural Netw. Learn. Syst. **24**(6), 845–867 (2013). https://doi.org/10.1109/TNNLS.2013.2247418

7. Hausknecht, M., Khandelwal, P., Miikkulainen, R., Stone, P.: HyperNEAT-GGP: a hyperNEAT-based atari general game player. In: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, pp. 217–224. ACM (2012). https://doi.org/10.1145/2330163.2330195
8. Kanerva, P.: Sparse distributed memory and related models, vol. 92. NASA Ames Research Center, Research Institute for Advanced Computer Science (1992)
9. Keller, P.W., Mannor, S., Precup, D.: Automatic basis function construction for approximate dynamic programming and reinforcement learning. In: Proceedings of International Conference on Machine Learning (2006). https://doi.org/10.1145/1143844.1143901
10. Li, L., Baker, T.E., White, S.R., Burke, K.: Pure density functional for strong correlations and the thermodynamic limit from machine learning. Phys. Rev. B **94**(24), 245129 (2016)
11. Li, W., Zhou, F., Meleis, W., Chowdhury, K.: Learning-based and data-driven TCP design for memory-constrained iot. In: 2016 International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 199–205. IEEE (2016). https://doi.org/10.1109/DCOSS.2016.8
12. Lin, S., Wright, R.: Evolutionary tile coding: an automated state abstraction algorithm for reinforcement learning. In: Abstraction, Reformulation, and Approximation (2010)
13. Mao, H., Netravali, R., Alizadeh, M.: Neural adaptive video streaming with pensieve. In: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, pp. 197–210. ACM (2017). https://doi.org/10.1145/3098822.3098843
14. Munos, R., Moore, A.: Variable resolution discretization in optimal control. Mach. Learn. **49**(2), 291–323 (2002)
15. Ratitch, B., Precup, D.: Sparse distributed memories for on-line value-based reinforcement learning. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS (LNAI), vol. 3201, pp. 347–358. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30115-8_33
16. Santamaría, J.C., Sutton, R.S., Ram, A.: Experiments with reinforcement learning in problems with continuous state and action spaces. Adapt. Behav. **6**(2), 163–217 (1997)
17. Smart, W.D., Kaelbling, L.P.: Practical reinforcement learning in continuous spaces. In: ICML, pp. 903–910 (2000)
18. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. Bradford Books (1998)
19. Whiteson, S., Taylor, M.E., Stone, P., et al.: Adaptive tile coding for value function approximation. University of Texas at Austin, Computer Science Department (2007)
20. Wu, C., Li, W., Meleis, W.: Rough sets-based prototype optimization in Kanerva-based function approximation. In: 2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), vol. 2, pp. 283–291. IEEE (2015). https://doi.org/10.1109/WI-IAT.2015.179
21. Wu, C., Meleis, W.: Adaptive Kanerva-based function approximation for multi-agent systems. In: Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems, vol. 3. pp. 1361–1364. International Foundation for Autonomous Agents and Multiagent Systems (2008). https://doi.org/10.1145/1402821.1402872

Predicting Drug Target Interactions Based on GBDT

Jiyun Chen, Jihong Wang, Xiaodan Wang, Yingyi Du, and Huiyou Chang(✉)

School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China
chenjy358@mail2.sysu.edu.cn, isschy@mail.sysu.edu.cn

Abstract. The research of the drug-target interactions (DTIs) is of great significance for drug development. Traditional chemical experiments are expensive and time-consuming. In recent years, many computational approaches based on different principles have been proposed gradually. Most of them use the information of drug-drug similarity and target-target similarity and made some progress. But the result is far from satisfactory. In this paper, we proposed machine learning method based on GBDT to predict DTIs with the IDs of both drug and protein, the descriptor of them, known DTIs and double negative samples. After gradient boosting and supervised training, GBDT construct decision trees for drug-target networks and generate precise model to predict new DTIs. Experimental results shows that Gradient Boosting Decision Tree (GBDT) reaches or outperforms other state-of-the-art methods.

Keywords: Drug-target interaction prediction · DTIs
Drug discovery · Machine learning · GBDT · DrugBank

1 Introduction

Drug research is the most widely used and most valuable field in the application of bioinformatics. It is costly and time consuming. Especially in the process of drug discovery, it takes more time and cost, which directly restricts the speed of new drug research.

A huge number of compounds and proteins are still under explored [1]. For instance, the latest release of DrugBank (version 5.0.11, released 2017-12-20) [2, 3] contains 10,571 drug entries including 2,338 approved small molecule drugs, 919 approved biotech drugs, 106 nutraceuticals and over 5,036 experimental drugs. Only 17587 non-repeating interactions linked by 6876 drugs and 4407 proteins are published, which means only 0.58% of the drug-target are found interacted [5].

Traditional computational approaches for identifying drug-target interactions can be divided into two categories, docking simulation methods and machine learning methods. Docking simulation methods utilize known 3D structure of targets to detect DTIs, with higher accuracy but at a higher cost. Moreover, the 3D structures of many targets are still difficult to obtain. Methods based on machine learning have attracted lots of attention. They can be divided into two subtypes: methods based on similarity and methods based on feature.

Since it is common acknowledged that similar drugs will interact with similar proteins and vice versa. With this assumption, methods based on similarity predict DTIs by computing similarity matrix of drug-drug or target-target with chemical structure of drug and protein sequence of target through particular similarity algorithm [4,6–8]. The drug and target similarities generate the chemical space of drugs and genomic space of targets, respectively, which explains the validity of similarity-based methods [9,10]. But these methods are computationally complex and do not fully make use of known DTIs and features of drug-target.

On the other hand, methods based on feature extract both drug and target features from molecular fingerprint of drug and sequence descriptor of target as well as known DTIs. Cao et al. [11,12] used three global descriptor (Composition, Transition and Distribution) to reflect biological properties of target sequence and encoded drug molecules with MACCS substructure fingerprint representing and then represent them as input feature vector for Random Forest Model. Ding et al. [13] employ substructure fingerprint of drug, physicochemical properties of target biology and drug-target relationship as input of Support Vector Machine model (SVM) and Feature Selection (FS). After taking full advantage of drug and target information, methods based on feature show high-performance.

Inspired by the above, we proposed a novel approach to identify DTIs by composing three types of information: (1) IDs of drug and target; (2) descriptor of drug and target; (3) DTIs. In this article, we randomly select drug-target samples from DrugBank dataset, which is known not interact, as negative samples. After feature selected, we adopt Gradient Boosting Decision Tree (GBDT) Model [14] to develop prediction system.

We examined the performance under different conditions: before or after FS, with or without IDs, different proportions of negative samples, etc. and compared our approach with state-of-the-art machine learning methods. Our experimental result shows that the proposed method achieves higher AUC (Area Under Curve) and prediction accuracy when feature selected, IDs retained and positive-negative proportion of the sample kept 1:2.

2 Methods

2.1 DTIs

In this paper, we only considered human protein. After removing duplicated data and nonhuman DTIs, we finally 12319 interactions, linked by 4950 drugs and 2313 human proteins. Moreover, DrugBank provides only positive samples. Hence we obtain negative samples by randomly select entries which is not exist in known DTI networks.

We use M = m_i (i = $1, 2, 3, \cdots$, n, n is the number of the sample) to represent feature vector matrix input by our predicting model, D = d_i (i = $1, 2, 3, \cdots$, n, n is the number of drugs in DTI network) to represent drug set, and T = t_i (i = $1, 2, 3, \cdots$, n, n is the number of targets in DTI network) for target set. M[i][DTI] = 1 means drug and target in i_{th} entity have interaction, 0 means no interaction or unknown.

2.2 Features of Drug

PyDPI [17] is an open source code which can help compute descriptors of drug and targets. We use it to download feature of drugs with drug id in DTIs obtained from DrugBank. Table 1 shows 609 dimension features we got.

Table 1. List of drug features

Features	Number of descriptors
Molecular constitutional descriptors	30
Topological descriptors	25
Molecular connectivity descriptors	44
Kappa descriptors	7
E-state descriptors	316
Moreau-Broto autocorrelation descriptors	32
Moran autocorrelation descriptors	32
Geary autocorrelation descriptors	32
Charge descriptors	25
Molecular property descriptors	6
MOE-type descriptors	60
Total	**609**

2.3 Features of Target

Like the descriptors of drug, we got 1819 dimension descriptors of target, as shown in Table 2:

2.4 GBDT

Gradient Boosted Decision Trees, GBDT, was first proposed by Friedman in 1999 [14]. At first, it was designed for CTR prediction in yahoo. As a supervised decision tree algorithm based on iterative accumulation, it constructs a set of weak learners (trees) and accumulates the results as the final predictive output. It is adaptable, easy to interpret. Furthermore, it produces highly accurate models [15].

GBDT takes two strategies to optimize the model.

Residual. The residual is actually the difference between real value and predicted result. GBDT first constructs a decision tree, and then get the residual value of "true value-predicted value", which is the learning target of the next tree. The loss function will reduce each round by fitting the residual.
Gradient descent algorithm. The gradient descent algorithm moves towards the negative gradient of the loss function each time, and finally get the minimum loss function.

Gradient Boosting algorithm is as follows:

1. GBDT first initializes loss function

$$F_0(x) = arg\,min_\rho \sum_{i=1}^{N} \Psi\left(y_i, \rho\right). \tag{1}$$

2. For m = 1 to M, do:

Table 2. List of protein features

Features	Number of descriptors
Amino acid composition	20
Dipeptide composition	400
Normalized Moreau-Broto autocorrelation	240
Moran autocorrelation	240
Composition	21
Transition	21
Distribution	105
Conjoint triad features	512
Sequence order coupling number	90
Quasi-sequence order descriptors	100
Pseudo amino acid composition	30
Amphiphilic pseudo amino acid composition	40
Total	**1819**

a. Calculate residual r$_{mi}$. It calculates the value of the negative gradient of the loss function in the current model as an estimate of the residual, which is the residual for the squared loss function or the approximation of the residual for the general loss function.

$$\widetilde{y_i} = -\left[\frac{\partial \Psi\left(y_i, F\left(x_i\right)\right)}{\partial F\left(x_i\right)}\right]_{F(X)=F_{m-1}(X)}, i = 1, \cdots, N. \tag{2}$$

b. Fit a regression tree for r$_{mi}$ to get the leaf area R$_{mj}$ of the m$_{th}$ tree. (j = $1, 2, \cdots$, J). (Estimated leaf area of regression tree, fitting the residual approximation) Fit a regression tree to the target rmi giving terminal regions.

c. Using linear search to estimate the value of the leaf node area to minimize the loss function.

$$\rho_m = arg\, min_\rho \sum_{i=1}^{N} \Psi\left(y_i, F_{m-1}\left(x_i\right) + \rho h\left(x_i; a_m\right)\right). \tag{3}$$

d. Update the regression tree

$$F_m\left(x\right) = F_{m-1}\left(x\right) + \rho_m h\left(x; a_m\right). \tag{4}$$

3. Output final model.

$$\hat{F}\left(x\right) = f_M\left(x\right). \tag{5}$$

GBDT is adaptable, easy to interpret. In addition, it produces highly accurate models and successfully used in a wide variety of applications. XGBOOST is a rather mature paradigm of machine learning base on GBDT [16]. It is a large-scale, distributed Gradient Boosting library that implements GBDT that implements algorithm of finding approximate split points and enables parallel computations. In this article, we build a GBDT model by XGBOOST.

Figure 1 is a schematic diagram of a decision tree generated by GBDT bases on DrugBank dataset.

2.5 Feature Engineering

We observed the distributions of 2430 features (id for drug, id for protein, 609 dimensions for drugs, 1819 dimensions for targets) in dataset and found that most of them display with long-tail distribution, as shown in Fig. 2. However, there are features that have single value in the whole dataset, which is definitely useless for our experiment. So we chose to discard these 290 features.

Furthermore, we also found that most features of few entities are null. 5 such valueless entities were abandoned.

GBDT is good at dealing with dense dataset so one-hot-encode is not needed. But it can only handle numeric values, which means ids in string type for drugs and targets must be label encoded.

So for dataset with 1:1 positive-negative proportion, the dimension becomes 2140 * 27314 from 2430 * 27319.

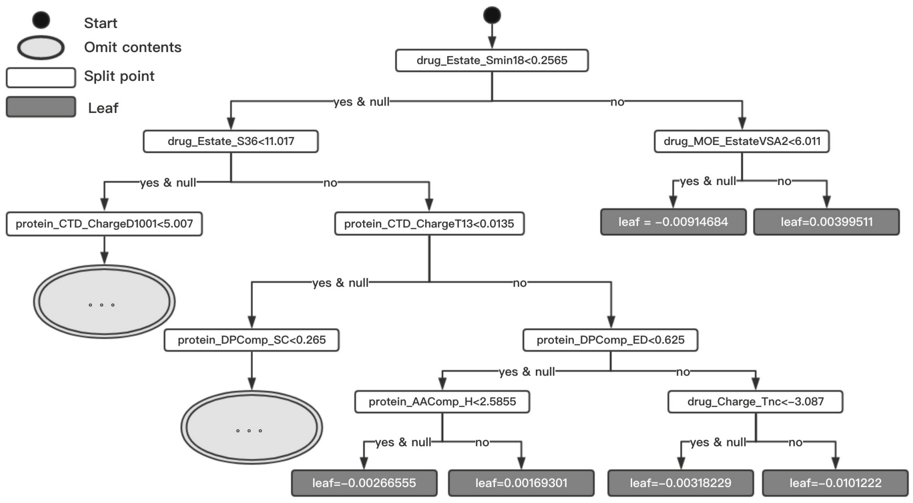


Fig. 1. A decision tree generated by GBDT based on DrugBank dataset

3 Experiment

There are several parameters to adjust in XGBOOST model. We change one parameter and keep others unchanged at a time. We finally got the best combination, max_depth = 10, min_child_weight = 4.7, gamma = 6, subsample = 0.88, colsample_bytree = 0.22, eta = 0.01, lambd = 0.31, seed27 and boost_round = 10000.

We applied AUC, accuracy and 10-fold validation to examine the performance of all models.

3.1 Validation

AUC (Area Under Curve) is defined as the area under the ROC (Receiver Operating Characteristic) curve [18].

ROC is the diagnostic ability of a binary classifier with regarding to different thresholds, while AUC curve displays true positive rate (sensitivity) versus false positive rate (1-specificity) at different values of thresholds. The sensitivity is the percentage of the test samples with ranks higher than a given threshold, whereas, specificity is the percentage the test samples that fall below the threshold.

Accuracy. After the prediction, each entry will have get a score. We set score $\geq$ split-point as positive and score $<$ split-point as negative. If the prediction equal to original label, then it is correct, vice versa. The accuracy is the result of dividing the number of correct predictions by the total number of test entries. In this paper, we set split-point equals 0.5.

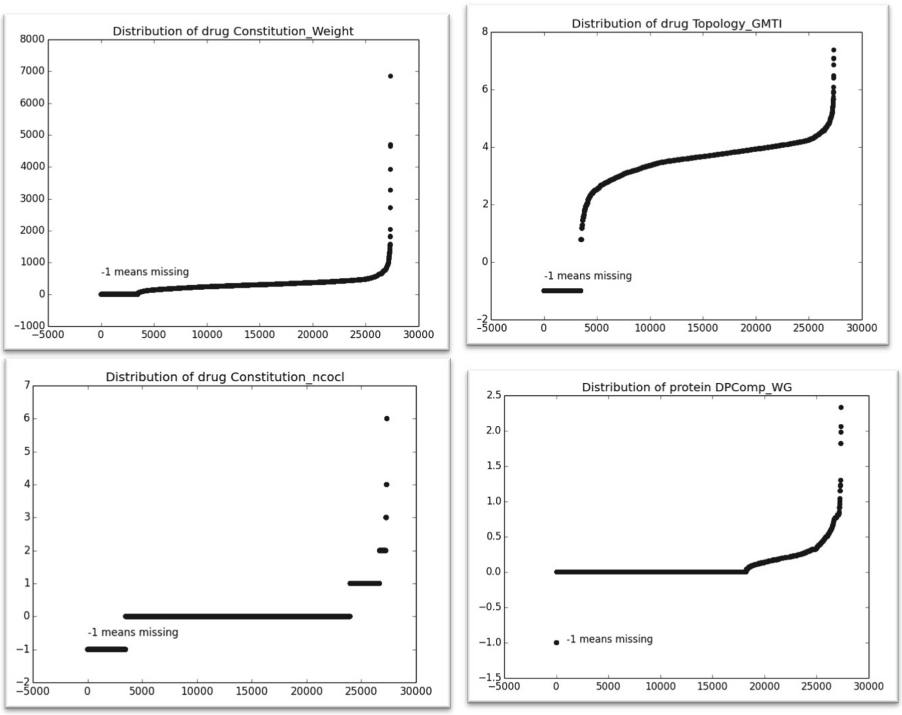


Fig. 2. Distribution of some features

K-Fold Cross Validation. Cross Validation is a common technique in statistics. One of the most popular method is the K-fold cross validation, which divides the dataset into K subsets with approximately the same size, and then utilize K-1 subsets as training data and keep the remaining subset as testing data [19]. Repeat it for K times where each subset has one chance to be the testing set. In order to make the result more convincing, we build every model based on 10-fold cross validation.

3.2 IDs of Drug and Target

We first using dataset containing drug id and target id to establish a GBDT model. Then rebuild another model with dataset without drug id and target id. Comparing the results of two models to examine the effect of the existence of the drug target ids on the prediction.

Table 3 in the bellow shows the results of prediction model when the dataset contains ids of drug and target or not. We can see that for a thousand iterations the AUC increases by about 0.0223 and accuracy increases by 0.0245, and for ten thousand interactions, the AUC increases by 0.0129 and accuracy increases by 0.165. In a word, the prediction model of using dataset with ids of drug and target has been significantly improved.

Table 3. Prediction result of dataset containing ids of drug and target or not

Dataset	boost_round	AUC	Accuracy
Without ids	1000	0.86395	0.789435
With ids	1000	0.88623	0.813928
Without ids	10000	0.89691	0.827088
With ids	10000	0.90979	0.843539

3.3 The Proportion of Negative Samples

Because our experimental dataset is rather small (less than 30,000 samples but up to 2,300 features), the predictions must be more accurate if we can add samples. The number of positive samples is consistent but the negatives can be changed. We added the number of negative samples, increasing the proportion of positive and negative samples to 1:2. In this way, number of sample raise.

Table 4 in the bellow shows the difference of prediction when the number of negative samples changed. The AUC and accuracy increased both in 1000 interactions and 10000 interactions. In addition, the AUC and accuracy increased by 0.014 and 0.045, respectively, when we add ids of drug and target and doubled the negative samples.

Table 4. Prediction result of different proportion of negative samples

Dataset positive vs negative	boost_round	AUC	Accuracy
1:1	1000	0.86395	0.789435
1:2	1000	0.86716	0.828258
1:1	10000	0.89691	0.827088
1:2	10000	0.89954	0.866856
Original	10000	0.89691	0.827088
Best combination	**10000**	**0.91095**	**0.871931**

3.4 Competing Methods

By observing the ids, number of negative samples and parameters, we can get the best combination, that is dataset with ids of drug and target, double negative samples. Then we compare this optimal model with six machine learning methods base on feature, Naive Bayes, Neural Net, SVM, Logistic Regression, Nearest Neighbors and Random Forest. Table 5 shows the result sorted by predicting performance of these methods. We can see Naive Bayes behaves worse, whereas Random Forest and our approach predict significantly better than the other methods.

We think this is because the sample has high rank and density matrix. Decision tree can help split point, which makes it more suitable for this type of dataset. And GBDT builds a new decision tree based on the goal of reducing the residual of the previous tree. It is more accurate than the last prediction on every iteration.

Figure 3 shows the Receiver Operating Characteristic Curve of our approach and other six methods. The area under the curve is called AUC. The higher the AUC, the better the model.

Table 5. Prediction result based on different machine learning methods

Method	AUC	Accuracy
Naive Bayes	0.54285	0.445622
Neural net	0.55611	0.544142
SVM	0.56119	0.597514
Logistic regression	0.62449	0.619996
Nearest neighbors	0.71011	0.663864
Random forest	0.87473	0.817584
Our approach	**0.91095**	**0.871931**

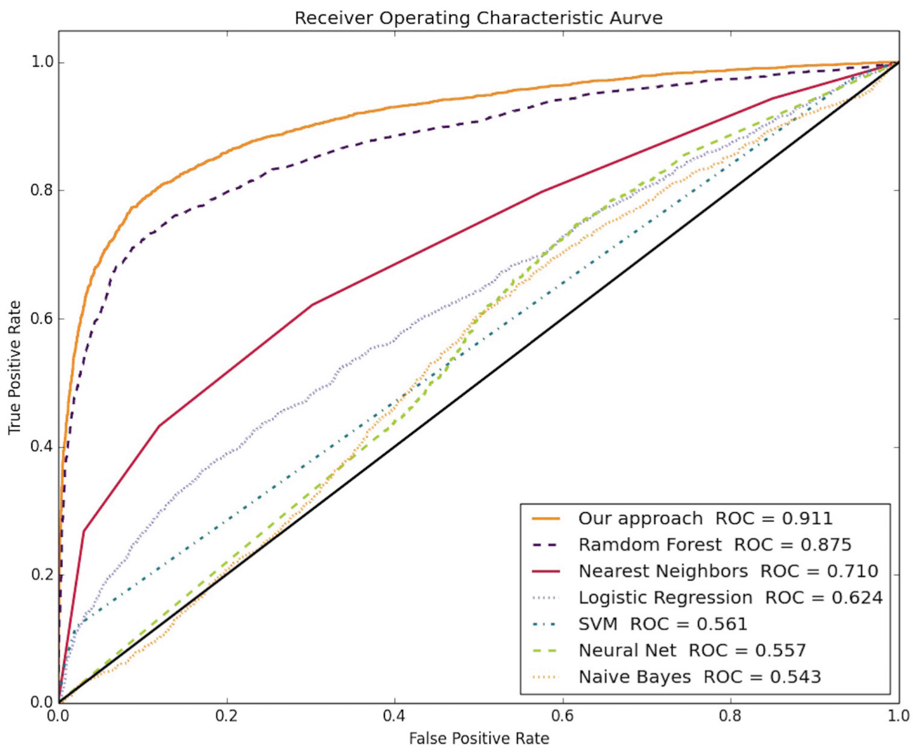


Fig. 3. ROC of different methods

4 Conclusion

Predicting DTI is especially a challenge for these three reasons. First, the database is not united. There are various datasets maintained by different organizations, DrugBank, ChEMBL, Benchmark, et al. Second, the volume of characteristics for drug and targets is very large, even larger than the number of known DTIs, which means the number of feature is greater than the number of samples. Third, all above dataset provides only positive DTIs, no negative DTIs [20]. So negative samples and unknown samples can only be marked as 0.

Most of the previous methods are based on similarity matrix and do not make good use of the information of drug target pairs. To improve drug target predictive performance, we have developed an approach based on GBDT that combines drug target ids, drug target descriptors, DTIs, and Feature selection. Different from other machine learning methods, GBDT has lower computational complexity and is more suitable for dense dataset. Our experimental results have proved that our method is superior to other competitive methods.

References

1. Santos, R., Ursu, O., Gaulton, A., et al.: A comprehensive map of molecular drug targets. Nat. Rev. Drug Discov. **16**(1), 19 (2017)
2. Law, V., Knox, C., Djoumbou, Y., et al.: DrugBank 4.0: shedding new light on drug metabolism. Nucleic Acids Res. **42**(Database issue), D1091 (2014)
3. Wishart, D.S., Feunang, Y.D., Guo, A.C., et al.: DrugBank 5.0: a major update to the DrugBank database for 2018. Nucleic Acids Res. **46**(D1), D1074–D1082 (2017)
4. Mei, J.P., Kwoh, C.K., Yang, P., et al.: Drugtarget interaction prediction by learning from local information and neighbors. Bioinformatics **29**(2), 238–245 (2013)
5. DrugBank. https://www.drugbank.ca/
6. Van, L.T., Nabuurs, S.B., Marchiori, E.: Gaussian Interaction Profile Kernels for Predicting Drug-Target Interaction. Oxford University Press, Oxford (2011)
7. Van, L.T., Marchiori, E.: Predicting drug-target interactions for new drug compounds using a weighted nearest neighbor profile. PLoS One **8**(6), e66952 (2013)
8. Gnen, M.: Predicting drugtarget interactions from chemical and genomic kernels using Bayesian matrix factorization. Bioinformatics **28**(18), 2304 (2012)
9. Zheng, X., Ding, H., Mamitsuka, H., et al.: Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1025–1033. ACM (2013)
10. Lu, Y., Guo, Y., Korhonen, A.: Link prediction in drug-target interactions network using similarity indices. BMC Bioinform. **18**(1), 39 (2017)
11. Cao, D.S., Zhang, L.X., Tan, G.S., et al.: Computational prediction of drugtarget interactions using chemical, biological, and network features. Mol. Inform. **33**(10), 669 (2014)
12. Cao, D.S., Liu, S., Xu, Q.S., et al.: Large-scale prediction of drug-target interactions using protein sequences and drug topological structures. Anal. Chim. Acta **752**(21), 1 (2012)
13. Ding, Y., Tang, J., Guo, F.: Identification of drug-target interactions via multiple information integration. Inf. Sci. **418**, 546–560 (2017)

14. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**(5), 1189–1232 (2001)
15. Ye, J., Chow, J.H., Chen, J., et al.: Stochastic gradient boosted distributed decision trees. In: ACM Conference on Information and Knowledge Management, pp. 2061–2064 (2009)
16. Chen, T., He, T., Benesty, M.: Xgboost: extreme gradient boosting. R Package Version 0.4-2, 1–4 (2015)
17. Cao, D.S., Liang, Y.Z., Yan, J., et al.: PyDPI: freely available python package for chemoinformatics, bioinformatics, and chemogenomics studies. J. Chem. Inf. Model. **53**(11), 3086–3096 (2013)
18. Lobo, J.M., Jimnez-Valverde, A., Real, R.: AUC: a misleading measure of the performance of predictive distribution models. Glob. Ecol. Biogeogr. **17**(2), 145–151 (2008)
19. Rodriguez, J.D., Perez, A., Lozano, J.A.: Sensitivity analysis of k-fold cross validation in prediction error estimation. IEEE Trans. Pattern Anal. Mach. Intell. **32**(3), 569–575 (2010)
20. Chen, X., Yan, C.C., Zhang, X., et al.: Drugtarget interaction prediction: databases, web servers and computational models. Brief. Bioinform. **17**(4), 696 (2016)

From Black-Box to White-Box: Interpretable Learning with Kernel Machines

Hao Zhang[1(✉)], Shinji Nakadai[1], and Kenji Fukumizu[2]

[1] NEC Corporation, Tokyo, Japan
h-zhang@cw.jp.nec.com
[2] The Institute of Statistical Mathematics, Tokyo, Japan

Abstract. We present a novel approach to interpretable learning with kernel machines. In many real-world learning tasks, kernel machines have been successfully applied. However, a common perception is that they are difficult to interpret by humans due to the inherent black-box nature. This restricts the application of kernel machines in domains where model interpretability is highly required. In this paper, we propose to construct interpretable kernel machines. Specifically, we design a new kernel function based on random Fourier features (RFF) for scalability, and develop a two-phase learning procedure: in the first phase, we explicitly map pairwise features to a high-dimensional space produced by the designed kernel, and learn a dense linear model; in the second phase, we extract an interpretable data representation from the first phase, and learn a sparse linear model. Finally, we evaluate our approach on benchmark datasets, and demonstrate its usefulness in terms of interpretability by visualization.

1 Introduction

Black-box models such as kernel machines and neural networks have proven to be highly accurate in many real-world applications. However, they are usually difficult to interpret by humans due to the inherent black-box nature. On the other hand, white-box models are naturally interpretable. For instance, generalized additive models (GAMs) [1] enable humans to interpret the relation between input features and output values thanks to the simple additive structure.

Here, a fundamental question to ask is: what does "interpretable" precisely mean? In this paper, based on the notion presented in [2], we refer to the term *interpretability* of predictive models as the following two aspects:

- Effects of input features on prediction of output values are understandable.
- Partial dependences of output values on input features are visualizable.

White-box models (e.g. GAMs) are easy to interpret by humans. Unfortunately, they are generally less accurate than black-box models. Accuracy is an important measure to evaluate predictive models. Interpretability, however, is as

© Springer International Publishing AG, part of Springer Nature 2018
P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 213–227, 2018.
https://doi.org/10.1007/978-3-319-96136-1_18

crucial as accuracy in many application domains such as life science [3], criminal justice [4] and marketing analytics [5]. For instance, when predicting customer churn rates, a marketer might wish to understand how the predictive model works as well, potentially for the purpose of promoting a campaign to avoid customer migration. In such application domains, white-box models are seemingly more favorable because of their relatively simple structures. However, Professor Leo Breiman argued that

> Interpretability is a way of getting information. But a model does not have to be simple to provide reliable information about the relation between predictor and response variable; neither does it have to be a data model [6].

This suggests that it is not necessary to have a preference for white-box models to pursue interpretability; instead, the key point is how to extract interpretable information from a model [7].

Motivated by this philosophy, we propose to interpret black-box models, in particular, kernel machines. Kernel machines such as support vector machines (SVMs) are a family of powerful nonparametric models [8,9]. The key idea is to *implicitly* map the input data to a high-dimensional space, and compute inner products in this space via a kernel function (a.k.a. the "kernel trick"). Owing to this implicit feature map, kernel machines are able to capture high-order interactions among features. However, the different effects of features and their interactions on the predictive model are not understandable, because everything is packed into the kernel function in a nontransparent way. Due to this black-box nature, standard kernel machines are difficult to interpret by humans.

In this paper, our objective is to construct interpretable kernel machines. Although kernel machines provide us with powerful predictive performances, the interpretability counterpart has not been thoroughly studied yet in the literature. The white-box RBF classifier [10] is an attempt to interpret kernel machines. Unfortunately, its storage and computation costs are extremely high because multiple kernel matrices have to be computed.

In contrast, we design a new kernel function based on random Fourier features (RFF) [11] to avoid computing kernel matrices. This kernel function consists of sub-kernels on *pairwise* features to capture nonlinear interactions. The corresponding feature map is explicitly built, so that efficient off-the-shelf linear algorithms can be exploited. With the designed kernel function, we develop a two-phase learning procedure:

– In the first phase, we explicitly map all feature pairs to a space produced by the designed kernel, and learn a dense linear model.
– In the second phase, we extract an interpretable data representation from the first phase, and learn a sparse linear model.

The reason why we focus on pairwise features is that we wish to capture main effects as well as two-way interactions in the extracted data representation. As for interactions, we only keep those involving two features due to the limitations of human perception and computer graphics for visualization.

The sparse model in the second phase admits an additive structure. This enables us to easily interpret different effects of features on the resulting model. Learning sparse models can be done typically by imposing sparsity-inducing regularizers to the optimization problem. The advantages of sparsity include model interpretability and also computational convenience.

2 Related Work

There have been increasing interests in developing interpretable models in recent years. The class of generalized additive models (GAMs) [1] relate the output values to a sum of univariate functions. Thanks to this additive structure, GAMs can be easily interpreted by humans. It is reported that fitting GAMs by using gradient boosting with regression trees can achieve high accuracy [12]. However, one of its potential weaknesses is the scalability issue due to the sequential manner of boosting.

More recently, post hoc methods such as LIME [13] have been developed to provide explanations for predictive models. These methods typically involve learning an extra model to interpret the fitted model of interest. On the other hand, our approach simultaneously performs prediction and interpretation.

Our approach is also closely related to wrapper methods of feature selection. Broadly, there are two categories of feature selection methods: wrapper and filter. Wrapper methods perform feature selection and prediction simultaneously; while filter methods select important features based on some statistical criterion, which is not directly connected to the learning algorithm.

Lasso [14] and ℓ_1-SVM [15] are two classic wrapper methods for regression and classification respectively. By using ℓ_1-regularizer, important features can be identified. Accordingly, the effects of these features on the resulting model can be interpreted by humans. Moreover, Lasso and ℓ_1-SVM are useful in large-scale problems thanks to the computational efficiency. However, a critical weakness is that nonlinear relations can not be captured. SVM recursive feature elimination (SVM-RFE) [16] is another wrapper method. It is originally designed for gene selection in life science. In SVM-RFE, irrelevant features are iteratively eliminated according to some ranking criterion during SVM training. Like Lasso and ℓ_1-SVM, the linear version of SVM-RFE can only capture linear relations. Although kernelized SVM-RFE is able to handle nonlinearity, it is not a scalable method any more. Sparse additive models (SpAM) [17] are a group of methods for nonparametric regression and classification. The optimization problem in SpAM is convex and can be efficiently solved by the back-fitting algorithm. SpAM is very effective for nonlinear feature selection. Nevertheless, since it does not assume any interaction among features, SpAM might perform poorly if interactions exist in the underlying model.

Our approach can be categorized as a scalable wrapper method of feature selection. However, our goal is different from that of feature selection. Specifically, our approach is to interpret and visualize different effects of input features on prediction, whereas feature selection is to choose a small subset of features that is sufficient to construct a predictive model.

3 Problem Setting

Suppose we have a dataset of N input-output data examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, where the vector $\mathbf{x}_i := \left(x_i^{(1)}, \ldots, x_i^{(D)}\right) \in \mathcal{X}$ is the i-th example with D features, and $y_i \in \mathcal{Y}$ is the corresponding prediction target. We have $\mathcal{Y} = \mathbb{R}$ for regression and $\mathcal{Y} = \{-1, +1\}$ for classification. A prediction of y is denoted by $\hat{y}$. Let $\mathbf{x}_i^{(p,q)} = \left(x_i^{(p)}, x_i^{(q)}\right)$ be a pairwise feature representation of $\mathbf{x}_i$, containing the p-th and q-th features.

3.1 Kernel Machines

Kernel machines are successful in many real-world tasks and mathematically well-founded. The core of kernel machines is the kernel function $\kappa \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$,

$$\kappa(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle_{\mathcal{H}},$$

where $\mathcal{H}$ is a high-dimensional Hilbert space and $\phi \colon \mathcal{X} \to \mathcal{H}$ is the corresponding feature map. The strength of kernel machines is that $\phi(\mathbf{x})$ does not need to be explicitly specified, because the inner product can be computed by κ in an relatively inexpensive way (a.k.a. "kernel trick"). There are several kernel functions successfully used in the literature, such as the polynomial kernel $\kappa_{\text{POL}}(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle + 1)^u, u \in \mathbb{N}$ and the Gaussian RBF kernel

$$\kappa_{\text{GAU}}(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}\|_2^2}{2\sigma^2}\right), \quad \sigma \in \mathbb{R}_{++}.$$

The Gaussian kernel is one of *shift-invariant* kernels, a wide class of kernel functions. A kernel function κ is shift-invariant if $\kappa(\mathbf{x}, \mathbf{z}) = \bar{\kappa}(\mathbf{x} - \mathbf{z})$, for some positive definite function $\bar{\kappa} \colon \mathcal{X} \to \mathbb{R}$. It is easy to see that the Gaussian kernel is shift-invariant, while the polynomial kernel is not. Unfortunately, in spite of its elegant properties, the kernel function κ is essentially a black box. Moreover, the associated optimization usually involves computing the $N \times N$ kernel matrix. This makes kernel machines unappealing in the large-scale scenario.

3.2 Random Fourier Features

Random Fourier features (RFF) [11] is an attractive and popular technique for improving scalability of shift-invariant kernels. The key insight is from a classical theorem [18] in harmonic analysis, which guarantees that

$$\kappa(\mathbf{x}, \mathbf{z}) = \bar{\kappa}(\Delta) = \int_{\mathbb{R}^D} e^{\sqrt{-1}\omega^{\mathsf{T}}\Delta} d\mathbb{P}(\omega).$$

Since $\bar{\kappa}(\Delta)$ is real-valued, we may have

$$\kappa(\mathbf{x}, \mathbf{z}) = \bar{\kappa}(\Delta) = \int_{\mathbb{R}^D} \cos(\boldsymbol{\omega}^{\mathsf{T}}\Delta) d\mathbb{P}(\boldsymbol{\omega}) = \mathbb{E}\left[\cos(\boldsymbol{\omega}^{\mathsf{T}}\Delta)\right].$$

Now build an *explicit* feature map $\hat{\phi}\colon \mathbb{R}^D \to \mathbb{R}^d$ as

$$\hat{\phi}(\mathbf{x}) := \sqrt{\frac{2}{d}} \left(\cos(\boldsymbol{\omega}_1^\mathsf{T}\mathbf{x}), \sin(\boldsymbol{\omega}_1^\mathsf{T}\mathbf{x}), \ldots, \cos(\boldsymbol{\omega}_{d/2}^\mathsf{T}\mathbf{x}), \sin(\boldsymbol{\omega}_{d/2}^\mathsf{T}\mathbf{x}) \right), \quad \{\boldsymbol{\omega}_i\}_{i=1}^{d/2} \overset{i.i.d.}{\sim} \mathbb{P}. \tag{1}$$

Then the corresponding kernel function $\hat{\kappa}\colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ can be written as

$$\hat{\kappa}(\mathbf{x}, \mathbf{z}) = \frac{2}{d} \sum_{i=1}^{d/2} \cos(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{x}) \cos(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{z}) + \sin(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{x}) \sin(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{z}) = \frac{2}{d} \sum_{i=1}^{d/2} \cos(\boldsymbol{\omega}_i^\mathsf{T} \Delta).$$

It is easy to see the condition $\mathbb{E}\left[\hat{\kappa}(\mathbf{x}, \mathbf{z})\right] = \kappa(\mathbf{x}, \mathbf{z})$ is satisfied. Hence $\hat{\kappa}(\mathbf{x}, \mathbf{z})$ is a sampling-based approximation of $\kappa(\mathbf{x}, \mathbf{z})$. Unlike the original feature map ϕ, $\hat{\phi}$ is relatively low-dimensional. This enables us to simply transform the input data with $\hat{\phi}$, and then exploit efficient linear algorithms to approximate the original nonlinear kernel machines.

4 Our Approach

In this section, we present our approach to learning with interpretable and scalable kernel machines (ISK). In ISK, we first design a new kernel function and then develop a two-phase learning procedure.

4.1 Kernel Function

We use an explicit feature map via RFF to approximate a certain kernel function. This kernel could be any shift-variant kernel. In this paper, we focus on the Gaussian kernel. By using RFF to approximate the Gaussian kernel, we have

$$\kappa_{\text{GAU}}(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle \approx \hat{\kappa}_{\text{GAU}}(\mathbf{x}, \mathbf{z}) = \langle \hat{\phi}(\mathbf{x}), \hat{\phi}(\mathbf{z}) \rangle,$$

where $\hat{\phi}$ is in the form of Eq. (1) and $\{\boldsymbol{\omega}_i\}_{i=1}^{d/2}$ are randomly sampled according to $\mathcal{N}(\mathbf{0}_D, \sigma^{-2}\mathbf{I}_D)$. Here, $\mathcal{N}(\mu, \boldsymbol{\Sigma})$ denotes the multi-variate Gaussian distribution with mean μ and covariance matrix $\boldsymbol{\Sigma}$.

Now we evaluate pairwise features $\mathbf{x}^{(p,q)}$ on $\hat{\kappa}_{\text{GAU}}$:

$$\hat{\kappa}_{\text{GAU}}(\mathbf{x}^{(p,q)}, \mathbf{z}^{(p,q)}) = \langle \hat{\phi}(\mathbf{x}^{(p,q)}), \hat{\phi}(\mathbf{z}^{(p,q)}) \rangle$$

with the feature map $\hat{\phi}\colon \mathbb{R}^2 \to \mathbb{R}^d$. Then we define our kernel function $\kappa_{\text{ISK}}\colon \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ as an average of sub-kernels on all distinct pairwise features:

$$\kappa_{\text{ISK}}(\mathbf{x}, \mathbf{z}) := \frac{1}{K} \sum_{p=1}^{D} \sum_{q>p}^{D} \hat{\kappa}_{\text{GAU}}(\mathbf{x}^{(p,q)}, \mathbf{z}^{(p,q)}) = \langle \phi_{\text{ISK}}(\mathbf{x}), \phi_{\text{ISK}}(\mathbf{z}) \rangle,$$

where $K = \binom{D}{2}$ is the total number of feature pairs, and the corresponding feature map $\phi_{\text{ISK}} \colon \mathbb{R}^D \to \mathbb{R}^{Kd}$ can be explicitly written as:

$$\begin{aligned}\phi_{\text{ISK}}(\mathbf{x}) := \sqrt{\frac{2}{Kd}} \Big(&\cos(\boldsymbol{\omega}_1^\mathsf{T}\mathbf{x}^{(1,2)}), \sin(\boldsymbol{\omega}_1^\mathsf{T}\mathbf{x}^{(1,2)}), \ldots, \\ &\cos(\boldsymbol{\omega}_{d/2}^\mathsf{T}\mathbf{x}^{(1,2)}), \sin(\boldsymbol{\omega}_{d/2}^\mathsf{T}\mathbf{x}^{(1,2)}), \ldots, \\ &\cos(\boldsymbol{\omega}_1^\mathsf{T}\mathbf{x}^{(D-1,D)}), \sin(\boldsymbol{\omega}_1^\mathsf{T}\mathbf{x}^{(D-1,D)}), \ldots, \\ &\cos(\boldsymbol{\omega}_{d/2}^\mathsf{T}\mathbf{x}^{(D-1,D)}), \sin(\boldsymbol{\omega}_{d/2}^\mathsf{T}\mathbf{x}^{(D-1,D)})\Big), \\ &\{\boldsymbol{\omega}_i\}_{i=1}^{d/2} \overset{i.i.d.}{\sim} \mathcal{N}(\mathbf{0}_2, \sigma^{-2}\mathbf{I}_2). \end{aligned} \tag{2}$$

There are also other kernel approximation techniques such as Nyström method [19,20]. The reason why we prefer RFF is that the sub-kernels in our kernel function are of similar forms, and RFF can be utilized for them in common.

4.2 Phase 1: Learning Dense Models

Since the feature map ϕ_{ISK} is explicitly built, we can easily transform the input data and exploit efficient linear algorithms to learn a dense (i.e. non-sparse) linear model

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi_{\text{ISK}}(\mathbf{x}) \rangle + w_0 \tag{3}$$

by solving the following convex optimization problem:

$$\min_{\mathbf{w}\in\mathbb{R}^{Kd}, w_0\in\mathbb{R}} \quad \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(f(\mathbf{x}_i), y_i) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2,$$

where $\mathcal{L}(f(\mathbf{x}), y)$ is a convex loss function and λ is a parameter controlling the importance of the regularization term.

For regression tasks where $\mathcal{Y} = \mathbb{R}$, taking the square loss $\mathcal{L}(f(\mathbf{x}), y) = \frac{1}{2}(y - f(\mathbf{x}))^2$ we obtain a standard ridge regression problem; for classification tasks where $\mathcal{Y} = \{-1, +1\}$, taking the hinge loss $\mathcal{L}(f(\mathbf{x}), y) = \max(0, 1 - yf(\mathbf{x}))$ we obtain a vanilla SVM problem.

4.3 Phase 2: Learning Sparse Models

In this phase, we first extract component representations from the fitted model in Eq. (3). To achieve this, we propose to expand trigonometric functions via Taylor series.

By the Taylor series of the cosine and sine functions around the point 0 (a.k.a. the Maclaurin series), we have

$$\begin{aligned}\cos(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{x}) &= \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!}(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{x})^{2n}, \\ \sin(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{x}) &= \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!}(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{x})^{2n+1}. \end{aligned} \tag{4}$$

Using the multinomial theorem to expand $(\boldsymbol{\omega}_i^\top \mathbf{x})^u$ as

$$(\boldsymbol{\omega}_i^\top \mathbf{x})^u = \sum_{p=1}^{D} (\omega_i^{(p)} x^{(p)})^u + \sum_{\substack{\sum_{l=1}^{D} r_l = u \\ r_l \neq u}} \binom{u}{r_1, \ldots, r_D} \prod_{1 \leq p \leq D} (\omega_i^{(p)} x^{(p)})^{r_p}. \tag{5}$$

Plugging Eq. (5) into Eq. (4) and let

$$C^{(p)}(\boldsymbol{\omega}_i, \mathbf{x}) := \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} (\omega_i^{(p)} x^{(p)})^{2n} = \cos(\omega_i^{(p)} x^{(p)}),$$
$$C^{(1,\ldots,D)}(\boldsymbol{\omega}_i, \mathbf{x}) := \cos(\boldsymbol{\omega}_i^\top \mathbf{x}) - \sum_{p=1}^{D} C^{(p)}(\boldsymbol{\omega}_i, \mathbf{x}).$$

Then it is easy to see that $\cos(\boldsymbol{\omega}_i^\top \mathbf{x})$ is separated into main effects and interaction effects as

$$\cos(\boldsymbol{\omega}_i^\top \mathbf{x}) = \sum_{p=1}^{D} C^{(p)}(\boldsymbol{\omega}_i, \mathbf{x}) + C^{(1,\ldots,D)}(\boldsymbol{\omega}_i, \mathbf{x}). \tag{6}$$

Similarly, we have

$$\sin(\boldsymbol{\omega}_i^\top \mathbf{x}) = \sum_{p=1}^{D} S^{(p)}(\boldsymbol{\omega}_i, \mathbf{x}) + S^{(1,\ldots,D)}(\boldsymbol{\omega}_i, \mathbf{x}). \tag{7}$$

In Eqs. (6) and (7), the terms $C^{(p)}(\boldsymbol{\omega}_i, \mathbf{x})$ and $S^{(p)}(\boldsymbol{\omega}_i, \mathbf{x})$ correspond to the main effects of the p-th feature, while $C^{(1,\ldots,D)}(\boldsymbol{\omega}_i, \mathbf{x})$ and $S^{(1,\ldots,D)}(\boldsymbol{\omega}_i, \mathbf{x})$ represent interactions. In our approach, only two-way interactions are included, because all of elements in the feature map ϕ_{ISK} are built on pairwise features.

We then rewrite the inner product $\langle \mathbf{w}, \phi_{\text{ISK}}(\mathbf{x}) \rangle$ in Eq. (3) as a sum, and expand each weighted element of $\phi_{\text{ISK}}(\mathbf{x})$ into three sub-components according to Eqs. (6) and (7). By combining these sub-components and centering the data, we obtain the $(D + K)$-dimensional component representation for each data example $\mathbf{x}$ as

$$\tilde{\mathbf{x}} := \left(\tilde{x}^{(1)}, \ldots, \tilde{x}^{(D)}, \tilde{x}^{(1,2)}, \ldots, \tilde{x}^{(D-1,D)} \right), \tag{8}$$

where the component $\tilde{x}^{(p)}$ is computed from the sum of sub-components related to the p-th feature, and the component $\tilde{x}^{(p,q)}$ is built based on the sum of sub-components that represent the interaction between the p-th and q-th features.

It is worth noting that centering the data should be done to ensure that each column has the zero mean and unit variance. Because we need to guarantee all the components are in the same scale so that the interpretation of their effects is not misleading.

With this interpretable component representation in Eq. (8), we learn a sparse linear model:

$$g(\tilde{\mathbf{x}}) = \boldsymbol{\beta}^{\mathsf{T}}\tilde{\mathbf{x}} + \beta_0 \tag{9}$$

by solving the following convex optimization problem:

$$\min_{\beta \in \mathbb{R}^{D+K}, \beta_0 \in \mathbb{R}} \quad \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(g(\tilde{\mathbf{x}}_i), y_i) + \tilde{\lambda}\rho\|\boldsymbol{\beta}\|_1 + \frac{\tilde{\lambda}(1-\rho)}{2}\|\beta\|_2^2,$$

where $\mathcal{L}(g(\tilde{\mathbf{x}}), y)$ is the loss function used in the first phase and $\tilde{\lambda}$ is a regularization parameter.

Here, the regularization term is a convex combination of ℓ_1-norm and ℓ_2-norm, with the parameter $\rho \in [0, 1]$ controlling their ratio. This is known as the *elastic net* regularizer [21]. Like ℓ_1-norm, elastic net can also yield sparse solutions of β, where only some of the coefficients in the fitted model are nonzero. Hence, components that are relevant for predicting the output values can be successfully identified. Only using ℓ_1-norm as the regularizer could be another choice for this sparse model. We instead use a more general regularizer elastic net because it allows us to balance the trade-off between sparseness and smoothness in practice.

4.4 Interpretation

Finally, we can interpret the resulting sparse model in Eq. (9) by checking component *importances* and *partial dependences* of output values on components in Eq. (8).

Since $\{\tilde{\mathbf{x}}_i\}_{i=1}^N$ are centered before fitting the sparse model in Eq. (9), the importance of the p-th component $\tilde{x}^{(p)}$ can be indicated simply by the magnitude of its coefficient in the fitted model, i.e. $|\beta^{(p)}|$. More specifically, $|\beta^{(p)}|$ represents the difference in the output value for each one-unit/category difference in $\tilde{x}^{(p)}$. Similarly, $|\beta^{(p,q)}|$ indicates the importance of the interaction component $\tilde{x}^{(p,q)}$.

Partial dependence plots (PDPs) are graphical renderings for predictive models [22,23]. By showing the dependence of output values on input features, PDPs are helpful for us to better understand the different effects of input features on the resulting model.

Now define the partial dependence function of the p-th component $\tilde{x}^{(p)}$ as a marginal average of g in Eq. (9):

$$g^{(p)}(\tilde{x}^{(p)}) := \mathbb{E}_{\tilde{\mathbf{x}}^{\setminus(p)}}\left[g(\tilde{x}^{(p)}, \tilde{\mathbf{x}}^{\setminus(p)})\right],$$

where $\tilde{\mathbf{x}}^{\setminus(p)}$ is the vector containing all of the elements of $\tilde{\mathbf{x}}$ except for $\tilde{x}^{(p)}$.

Although $g^{(p)}$ cannot be directly accessed, empirical estimation is possible:

$$\bar{g}^{(p)}(\tilde{x}^{(p)}) := \frac{1}{N}\sum_{i=1}^{N} g(\tilde{x}^{(p)}, \tilde{\mathbf{x}}_i^{\setminus(p)}) = \beta^{(p)}\tilde{x}^{(p)} + \frac{1}{N}\sum_{i=1}^{N}\langle \beta^{\setminus(p)}, \tilde{\mathbf{x}}_i^{\setminus(p)}\rangle. \tag{10}$$

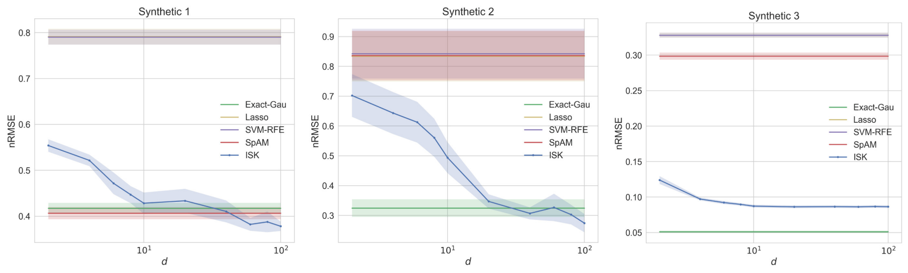


Fig. 1. Results on synthetic data. Normalized root mean square errors (nRMSEs) are averaged on 10 random splits of data. Means and standard errors of nRMSEs are reported. "Exact-Gau" stands for "Kernel ridge regression with the exact Gaussian kernel". The lines of Lasso and SVM-RFE overlap each other because their performances are comparable.

Similarly, the empirical partial dependence function for the interaction component $\tilde{x}^{(p,q)}$ can be written as:

$$\bar{g}^{(p,q)}(\tilde{x}^{(p,q)}) := \beta^{(p,q)}\tilde{x}^{(p,q)} + \frac{1}{N}\sum_{i=1}^{N}\langle \beta^{\backslash(p,q)}, \tilde{\mathbf{x}}_i^{\backslash(p,q)}\rangle. \tag{11}$$

These partial dependence functions can be easily visualized for interpretation of the resulting model.

5 Experiments

In this section, we describe our experimental design and report comparison results. We first conduct synthetic experiments for illustration; then we use several large benchmark datasets for performance comparison; finally, we demonstrate the usefulness of our approach in terms of interpretability by visualization.

We compare our approach with several existing interpretable and scalable *wrapper* methods, including Lasso [14], ℓ_1-SVM [15], SVM-RFE [16] and SpAM [17,24].

5.1 Synthetic Data

To illustrate the properties of our approach, we first generate three synthetic datasets. We generate $N = 1000$ data examples of dimension $D = 5$ for the first two synthetic datasets. Data examples are randomly sampled by following the standard Gaussian distribution. For the third synthetic dataset, we generate $N = 1000$ data examples of dimension $D = 10$, where all of the columns are randomly sampled from the uniform distribution $\mathcal{U}[0,1]$ except for $x^{(4)}, x^{(5)}, x^{(8)}, x^{(10)} \sim \mathcal{U}[0.6,1]$. Then three underlying true models are generated as:

$$
\begin{aligned}
f_1^*(\mathbf{x}) =& -2\sin(2x^{(1)}) + (x^{(2)})^2 + x^{(3)} + \exp(-x^{(4)}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0,1).\\
f_2^*(\mathbf{x}) =& x^{(1)}\exp(2x^{(2)}) + (x^{(3)})^2 + \epsilon, \quad \epsilon \sim \mathcal{N}(0,1).\\
f_3^*(\mathbf{x}) =& \pi^{x^{(1)}x^{(2)}}\sqrt{2x^{(3)}} - \sin^{-1}(x^{(4)}) + \log(x^{(3)} + x^{(5)}) - \frac{x^{(9)}}{x^{(10)}}\sqrt{\frac{x^{(7)}}{x^{(8)}}} - x^{(2)}x^{(7)}.
\end{aligned}
$$

Synthetic 1 has a typical additive structure of main effects, which is originally used in [17]; the true model in Synthetic 2 includes a pairwise interaction, and it is generated in [25]; Synthetic 3 involves more complex interactions, and it is used in [26].

We also test kernel ridge regression with the Gaussian kernel (Exact-Gau) for reference. We evaluate all the methods in terms of prediction performance, by using the normalized root mean square error (nRMSE) as the metric. For our approach, we vary values of d in Eq. (2) to see how the performance changes.

Each dataset is randomly split into training data (80%) and test data (20%). We perform 5-fold cross-validation on training data for tuning parameters. The model is finally refitted and evaluated on test data. The entire procedure is repeated for 10 times.

Figure 1 shows the comparison results on synthetic datasets. We can observe that the performance of our approach gets better and better as d increases in all the three cases. This is consistent with the theoretical properties of RFF. When d is large enough, our approach can perform comparably well as Exact-Gau.

The performance of SpAM is extremely good on Synthetic 1 (even better than Exact-Gau). This is because SpAM assumes an additive structure of main effects, which is exactly how we generate data in Synthetic 1. However, when we add interaction effects in Synthetic 2 and 3, SpAM cannot perform well any more. Generally, Lasso and SVM-RFE have comparable worst performances on all the three synthetic datasets. This is not surprising because Lasso and SVM-RFE cannot capture nonlinear relations. On the other hand, Exact-Gau achieves high performance in all the three cases, owing to its capability of modeling infinite-order nonlinear interactions. However, as a standard kernel method, Exact-Gau is difficult to scale up to larger datasets because its computational complexity is quadratic in the number of data example N, while that of our approach is linear in N.

5.2 Benchmark Data

We conduct comparison experiments using public benchmark data as listed in Table 1, including five regression and five classification datasets.

Most of these datasets are from the application domains where interpretability is highly required, such as "Parkinsons" from life science and "Bank" from marketing analytics. We decided not to include Exact-Gau in benchmark experiments because of its high costs of storage and computation. For regression tasks, we use the normalized root mean square error (nRMSE) as the metric; for classification tasks, we use the error rate as the metric. Error rate is calculated as the proportion of incorrect predictions of the class label.

Table 1. Specifications of benchmark data. Five for regression and five for classification.

Dataset	# of examples	# of features	% of pos.
Parkinsons	5875	20	-
Pumadyn	8192	8	-
CalHousing	20640	8	-
Kin40k	40000	8	-
Protein	45730	9	-
Spambase	4601	57	39.40
Eye	14980	14	44.88
Credit	30000	23	22.12
Bank	41188	20	11.27
Cod-RNA	59535	8	33.33

Table 2. Results on benchmark datasets. Normalized root mean square errors (nRMSEs) and error rates are averaged on 10 random splits of data. Means and standard deviations of nRMSEs and error rate (%) are reported.

	Lasso/ℓ_1-SVM	SVM-RFE	SpAM	ISK
Parkinsons	88.90 (1.41)	88.87 (1.56)	82.88 (1.34)	**56.49 (2.33)**
Pumadyn	79.06 (1.16)	79.02 (1.14)	74.90 (0.86)	**61.74 (3.70)**
CalHousing	60.95 (1.02)	60.98 (1.11)	59.08 (1.06)	**57.03 (1.50)**
Kin40k	100.09 (0.75)	100.10 (0.74)	97.38 (0.67)	**82.98 (2.33)**
Protein	84.77 (0.48)	84.85 (0.54)	82.41 (0.37)	**80.05 (0.99)**
Spambase	7.82 (1.01)	7.75 (0.82)	6.91 (1.48)	**6.38 (0.89)**
Eye	39.20 (1.49)	39.99 (1.36)	34.78 (2.06)	**19.92 (3.40)**
Credit	19.13 (1.00)	19.87 (1.44)	19.89 (0.20)	**17.90 (0.39)**
Bank	9.75 (0.23)	9.55 (0.35)	9.44 (0.15)	**9.15 (0.33)**
Cod-RNA	6.16 (0.12)	6.18 (0.13)	5.76 (0.44)	**4.96 (0.21)**

Table 2 shows the comparison results on benchmark datasets. From the results, we can see our approach yields the smallest prediction errors and performs better than other methods on all the datasets. SpAM has the second best performance in general. The reason why our approach performs better than SpAM might be that our approach is able to detect interaction effects between features. Lasso/ℓ_1-SVM and SVM-RFE perform the worst among all the methods. This might be due to the linearity of Lasso/ℓ_1-SVM and the usage of linear SVM in SVM-RFE instead of the kernel-based version for the purpose of scalability.

5.3 Visualization

Finally, we demonstrate how to interpret the resulting model by checking component importances and visualizing partial dependences. We compute these quantities based on the sparse model in Eq. (9) fitted on the whole dataset. Due to the space limitation, we only show the figures from the "CalHousing" data [27].

Table 3. Features in "CalHousing".

Name	Description
MedInc	Median income
HouseAge	Housing median age
AveRooms	Average number of rooms
AveBedrms	Average number of bedrooms
Population	Population in each block group
AveOccup	Average occupancy in each house
Latitude	Geographic coordinate (north-south)
Longitude	Geographic coordinate (east-west)

"CalHousing" consists of $N = 20460$ examples, each of which represents the aggregated data of a block group in California from the 1990 Census. The prediction target is the median house value in each block group, and the features ($D = 8$) include demographic, geographical and residential information. The feature information of "CalHousing" is summarized in Table 3.

We perform our approach on the "CalHousing" data. We first check component importances by looking at the coefficients of the fitted sparse model in Eq. (9). Then we calculate the *relative* importance for each component. Since the components with low importances do not have significant effects on the resulting model, we are more interested in top components (e.g. those with relative importance ≥ 0.2).

Figure 2 shows the ranking of top components on the "CalHousing" data. The main effect of MedInc is identified as the most important component for predicting house value, followed by location-related components: the main effects of Latitude and Longitude, together with the interaction between them. AveOccup and AveRooms have less than half the importance of MedInc, so they are somehow relatively unimportant among these top 6 components.

We then calculate the partial dependences of house value on these top ranking components by using Eqs. (10) and (11).

The first three plots in Fig. 3 show the partial dependences on main effect components. We can observe that the partial dependence of house value is generally monotonic increasing as MedInc increases over the main body of data. House value has a linearly decreasing partial dependence on AveOccup. The partial dependence of house value on AveRooms is a non-monotonic, where the

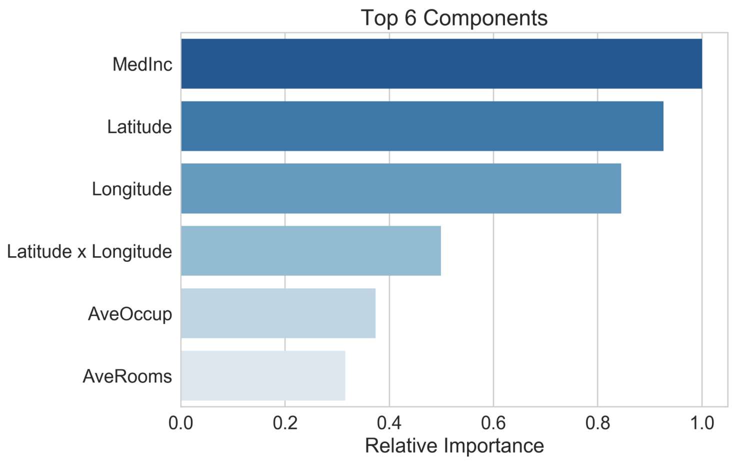


Fig. 2. Ranking of top 6 components (relative importance ≥ 0.2) on CalHousing data.

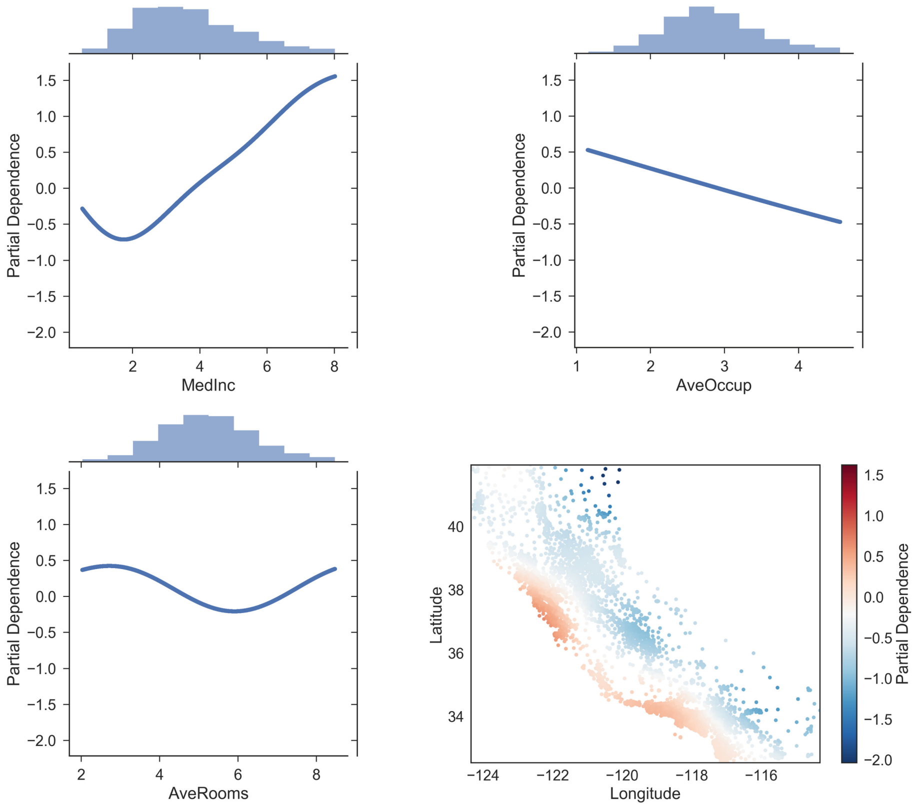


Fig. 3. Partial dependence plots on “CalHousing”.

minimum is approximately at 6 rooms, roughly corresponding to the highest proportion in the data.

The partial dependences on location-related components are rather interesting. The last plot in Fig. 3 shows the partial dependence on the interaction between Latitude and Longitude. House value is seen to largely depend on this interaction effect along the coastline of California, especially in the area around (37, −122), which is near the Bay Area of California.

6 Conclusion

We proposed a novel approach to interpretable learning with kernel machines. We first designed a new kernel function based on random Fourier features for the scalability purpose, and then developed a learning procedure including two phases: in the first phase, we map all pairs of features to an explicit feature space produced by the designed kernel to learn a dense linear model; in the second phase, we use a component extraction trick to represent the original data in an interpretable way for learning a sparse linear model. Experimental results on several large benchmark datasets showed the predictive power of our approach, and the visualization demonstrates its usefulness for interpretability.

References

1. Hastie, T.J., Tibshirani, R.J.: Generalized Additive Models. Chapman & Hall/CRC, Boca Raton (1990)
2. Lipton, Z.C.: The mythos of model interpretability. In: ICML 2016 Workshop on Human Interpretability in Machine Learning (WHI2016) (2016)
3. Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., Elhadad, N.: Intelligible models for healthcare: predicting pneumonia risk and hospital 30-day readmission. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD2015), pp. 1721–1730. ACM (2015)
4. Zeng, J., Ustun, B., Rudin, C.: Interpretable classification models for recidivism prediction. J. Royal Stat. Soc. Ser. A (Stat. Soc.) **180**, 689–722 (2016)
5. Letham, B., Letham, L.M., Rudin, C.: Bayesian inference of arrival rate and substitution behavior from sales transaction data with stockouts. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD2016), pp. 1695–1704. ACM, New York (2016)
6. Breiman, L.: Statistical modeling: the two cultures (with comments and a rejoinder by the author). Stat. Sci. **16**(3), 199–231 (2001)
7. Chang, B.H.W.: Kernel machines are not black boxes - on the interpretability of kernel-based nonparametric models. Ph.D. thesis, University of Toronto (2014)
8. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
9. Vapnik, V., Golowich, S.E., Smola, A.J.: Support vector method for function approximation, regression estimation and signal processing. In Jordan, M.I., Petsche, T. (eds.) Advances in Neural Information Processing Systems (NIPS 1996), vol. 9, pp. 281–287. MIT Press (1997)

10. Van Belle, V., Lisboa, P.: White box radial basis function classifiers with component selection for clinical prediction models. Artif. Intell. Med. **60**(1), 53–64 (2014)
11. Rahimi, A., Recht, B.: Random features for large-scale kernel machines. In: Platt, J.C., Koller, D., Singer, Y., Roweis, S.T. (eds.) Advances in Neural Information Processing Systems (NIPS 2007), vol. 20, pp. 1177–1184. Curran Associates, Inc. (2008)
12. Lou, Y., Caruana, R., Gehrke, J.: Intelligible models for classification and regression. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2012), pp. 150–158. ACM (2012)
13. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?": explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016), pp. 1135–1144. ACM (2016)
14. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. Royal Stat. Soc. Ser. B Methodol. **58**, 267–288 (1996)
15. Zhu, J., Rosset, S., Tibshirani, R., Hastie, T.J.: 1-norm support vector machines. In: Thrun, S., Saul, L.K., Schölkopf, P.B. (eds.) Advances in Neural Information Processing Systems (NIPS 2003), vol. 16, pp. 49–56. MIT Press (2004)
16. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Mach. Learn. **46**(1–3), 389–422 (2002)
17. Ravikumar, P., Lafferty, J., Liu, H., Wasserman, L.: Sparse additive models. J. Royal Stat. Soc. Ser. B (Stat. Methodol.) **71**(5), 1009–1030 (2009)
18. Bochner, S.: Lectures on Fourier Integrals. Princeton University Press, Princeton (1959)
19. Williams, C.K.I., Seeger, M.: Using the Nyström method to speed up kernel machines. In: Leen, T.K., Dietterich, T.G., Tresp, V. (eds.) Advances in Neural Information Processing Systems (NIPS 2000), vol. 13, pp. 682–688. MIT Press (2001)
20. Drineas, P., Mahoney, M.W.: On the Nyström method for approximating a Gram matrix for improved kernel-based learning. J. Mach. Learn. Res. **6**, 2153–2175 (2005)
21. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. J. Royal Stat. Soc. Ser. B (Stat. Methodol.) **67**(2), 301–320 (2005)
22. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**(5), 1189–1232 (2001)
23. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer Series in Statistics, 2nd edn. Springer, New York (2009). https://doi.org/10.1007/978-0-387-84858-7
24. Zhao, T., Liu, H.: Sparse additive machine. In: Lawrence, N.D., Girolami, M.A. (eds.) Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS 2012), pp. 1435–1443 (2012)
25. Yamada, M., Jitkrittum, W., Sigal, L., Xing, E.P., Sugiyama, M.: High-dimensional feature selection by feature-wise non-linear lasso. Neural Comput. **26**(1), 185–207 (2014)
26. Lou, Y., Caruana, R., Gehrke, J., Hooker, G.: Accurate intelligible models with pairwise interactions. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2013), pp. 623–631. ACM (2013)
27. Pace, R.K., Barry, R.: Sparse spatial autoregressions. Stat. Probab. Lett. **33**(3), 291–297 (1997)

A Two-List Framework for Accurate Detection of Frequent Items in Data Streams

David Vengerov(✉)

Oracle Labs, Belmont, CA 94002, USA
david.vengerov@oracle.com

Abstract. The problem of detecting the most frequent items in large data sets and providing accurate frequency estimates for those items is becoming more and more important in a variety of domains. We propose a new two-list framework for addressing this problem, which extends the state-of-the-art Filtered Space-Saving (FSS) algorithm. An algorithm called FSSA giving an efficient array-based implementation of this framework is presented. An adaptive version of this algorithm is also presented, which adjusts the relative sizes of the two lists based on the estimated number of distinct keys in the data set. Analytical comparison with the FSS algorithm showed that FSSA has smaller expected frequency estimation errors, and experiments on both artificial and real workloads confirm this result. A theoretical analysis of space and time complexity for FSSA and its benchmark algorithms was performed. Finally, we showed that FSS2L framework can be naturally parallelized, leading to a linear decrease in the maximum frequency estimation error.

Keywords: Data mining · Frequent items · Skew detection · Space-Saving

1 Introduction

The problem of detecting the most frequent items in large data sets and providing accurate frequency estimates for those items arises in many areas:

- Internet providers need to know the most frequent destinations in order to manage traffic and service quality.
- Social network companies need to find the most frequent interactions among the users in order to extract information about connections and relations between individuals.
- Retail companies need to know the most common products purchased by each customer in order to better classify the customer and design appropriate marketing campaigns.
- An increasing number of businesses are finding it useful to monitor the most frequent topics in news streams or social media so as to adjust their business decisions based on these topics.
- Database engines can optimize query plans depending on the degree of skew in the underlying data and can give separate treatment to the most frequent items.

In order to be practical, any algorithm that tackles this problem has to perform its computations using limited memory, usually orders of magnitude less than the size of the data set. Another key practical requirement is performing all computations in one pass – without this feature, an algorithm would not be useful for very large data sets or data streams.

Several algorithms that meet these requirements have been proposed in the literature, the most well-known ones being Frequent [3], Lossy Counting (LC) [6] and Space-Saving (SS) [8]. The Frequent algorithm was shown to be much less accurate than either LC or SS (see, for example, [1, 7]). The Space-Saving (SS) algorithm was shown (e.g., [1, 7]) to be more accurate than Lossy Counting (LC). During our literature search, we found an improved version of SS called Filtered Space-Saving (FSS) [5], which was shown to be more accurate than SS, both in [5] and in our experiments in Sect. 5.

However, we discovered that the FSS algorithm suffers from the fact that infrequently observed items can still get onto the list of items tracked by FSS and displace from it many of the frequent items that we would like to keep. In order to avoid this problem, we extended the FSS algorithm by keeping information about the frequently observed items in a special protected list, which prevents that information from being displaced by the infrequent items. We refer to this two-list framework as FSS2L.

The rest of this paper is organized as follows. Section 2 describes the previously developed SS and FSS algorithms and presents a novel analysis of the expected behavior of the FSS algorithm. Section 3 presents the FSS2L framework and analyzes its expected behavior. It then describes an efficient implementation of this framework using a single array, which we called FSSA algorithm. Finally, it presents an algorithm for adapting the relative sizes of the two lists used in the FSS2L framework and discusses the impact of such adaptation. Section 4 presents analysis of the space and time complexity of FSSA and its benchmark algorithms. Section 5 evaluates these algorithms on both artificial and real distributions of item keys. Section 6 summarizes the results and explains how the FSSA algorithm can be parallelized.

2 Relevant Prior Algorithms

2.1 Space-Saving Algorithm

The Space-Saving (SS) algorithm [8] stores the estimated frequency of each monitored key in a hash map with a maximum size m. If the observed key i has a corresponding entry in the hash map, then its frequency estimate f_i is incremented. Otherwise, an entry is added to the hash map with the estimated frequency $f_{min} + 1$, where f_{min} is the minimum frequency among all the keys currently stored in the hash map. When the number of entries in the hash map exceeds m, the entry corresponding to the key with the smallest estimated frequency f_{min} is removed.

By assigning the frequency $f_{min} + 1$ to each newly inserted key, the SS algorithm never underestimates a key's frequency. This is so because a newly observed key could have been previously observed at most f_{min} times, then removed from the hash map (possibly when the previous key was inserted), and was just observed again. Since a key that was never observed before will be inserted with a frequency $f_{min} + 1$, the maximum possible error Δ_i in the frequency estimate for a key i satisfies $0 \leq \Delta_i \leq f_{min}$. The value f_{min} will grow at the fastest rate if all observed keys are distinct, in which case f_{min} will be incremented by 1 after m keys are observed. Therefore, $f_{min} \leq N/m$, and the final worst-case accuracy guarantee for SS becomes $0 \leq \Delta_i \leq N/m$. A formal proof of this fact is given in [8].

2.2 Filtered Space-Saving Algorithm

The Filtered Space-Saving (FSS) algorithm [5] improves the accuracy of the SS algorithm by storing fewer keys in the hash map and allocating the "saved" space to an array H that holds hashed key values. Instead of always adding a new entry to the hash map when a key is observed for which no entry exists, the key is first hashed, uniformly, into the set $\{1, 2, \ldots, sizeH\}$, where $sizeH$ - is the number of cells in H. If the key is hashed to k and $H[k]$ is smaller than $f_{min} - 1$, then $H[k]$ is increased by 1 and no further action is taken. Otherwise, a new entry is created in the hash map with this key, its estimated frequency f_i is set to $H[k] + 1$ and its maximum error Δ_i is set to $H[k]$. When the number of keys in the hash map exceeds the preset upper bound, a key with frequency equal to f_{min} is removed, and if $H[j] < f_{min}$, where j is the value to which the removed key hashes, then $H[j]$ is set to f_{min} so that it would still represent the maximum overestimation error for any newly inserted key that hashes to cell j (in case the key with frequency f_{min} that was just removed is the next key to be observed again).

FSS has the following worst-case accuracy guarantee: $0 \leq \Delta_i \leq H[k] \leq f_{min} \leq N/M$, where M is the number of elements it stores in the hash map. These guarantees can be easily inferred from the operations of FSS. Since Δ_i for each newly inserted key i that hashes to cell k is set to $H[k]$, and $H[k]$ can still grow over time, we get $0 \leq \Delta_i \leq H[k]$. The values stored in H are incremented only when $H[k] + 1 < f_{min}$ for the observed key or when a key is evicted and its corresponding $H[j] < f_{min}$, implying that $H[k]$ is always $\leq f_{min}$ for every k. Finally, let S_0 be the sum of the frequencies of all keys monitored by FSS and note that every observed key either increments S_0 by 1 or leaves it unchanged. Since N keys are observed, $N \geq S_0 \geq Mf_{min}$, which implies $f_{min} \leq N/M$.

Since FSS stores fewer keys in its hash map than SS, the worst-case error for FSS derived above is larger than that for SS. However, such an error can only be observed if all keys happen to hash to the same cell in the hash array, which has a negligible probability for any real hashing scheme whose goal is to spread the keys uniformly over all cells in the hash array.

2.3 Efficiently Finding the Smallest Frequency Key

In order to implement efficiently all frequency estimation algorithms built on top of SS, one needs to be able to quickly find the key with the smallest frequency f_{min} when it needs to be evicted from the hash map. In order to perform this eviction in $O(1)$ time, the authors of the SS algorithm came up with the "Stream Summary" data structure [8] for storing information about the monitored keys.

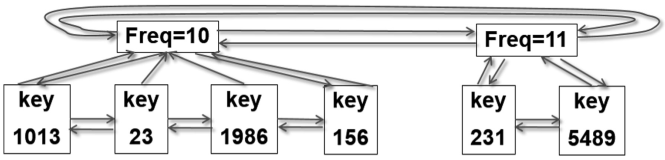


Fig. 1. The “modified” Stream Summary linked list data structure used for storing keys in our implementation of SS and FSS.

In this data structure, each element in the top-level double linked list is a “bucket” with a specified frequency F_b (either 10 or 11 in Fig. 1), which contains its own linked list of nodes that stores keys that have estimated frequency F_b. The nodes are always added to the end of the list inside each bucket. If the total number of keys tracked by Stream Summary has exceeded a prespecified threshold, then the first node in the first bucket is removed. Each “bucket” maintains pointers to the first node and to the last node on its list, which makes insertion and removal operations take $O(1)$ time in the linked list. When a key that is present in the hash map with an estimated frequency f_i is observed in the data set, it is moved to the end of the linked list in the bucket with $F_b = f_i + 1$, and if such a bucket does not exist, a new bucket is created (and is inserted into a proper location in the list of buckets so as to still keep it sorted by frequency) and the node with the observed key is added to the list of nodes in that bucket.

In order to perform the removal of a node from its current location in $O(1)$ time, each node can also maintain a pointer to the previous node in the list, making the list of nodes doubly linked (otherwise, a linear search will be needed in order to find the node before the one being removed, so that it could then be linked with the node after the one being removed). This modified version of the Stream Summary was used in the experiments described in Sect. 5 and is shown in Fig. 1.

We also used a suggestion that was made in [8] (and explained in more detail in [2]) for the hash map to hold pointers into the Stream Summary, so that no linear search would be needed in order to find specific keys in it. This implementation requires storing 2 numbers in the hash map for each monitored key (the key and the pointer to the Stream Summary node) and storing 4 numbers in the corresponding node: the key and the pointers to the parent bucket, to the previous node and to the next node.

3 Proposed Algorithms

3.1 FSS2L Framework

The general idea of the proposed FSS2L framework is to split the hash map entries into two conceptual lists: List0 where new keys are first placed and List1 to which keys that are considered to be frequent are promoted from List0. Keys in List0 are sorted by their estimated frequency, while those in List1 are sorted by their “hits” value – the number of times there were observed since they were first added to List0. The hits value of each monitored key i can be computed dynamically from its estimated frequency and maximum error, as will be explained below.

The operations of this FSS-two-list (FSS2L) framework are shown in Fig. 2. When a new key i is observed and is found in List1, its estimated frequency is incremented and it is moved closer to the top of List1 in order to still keep that list sorted by hits. If the observed key i is found in List0, then it is upgraded to List1 if $h_i \geq h_{min} - 1$ (where h_{min} is the minimum number of hits for any element in List1) and its estimated frequency is incremented. Otherwise, the estimated frequency f_i for this key is incremented and, if needed, the key is moved closer to the top of List0. If the key is upgraded and if the number of elements m_1 in List1 became greater than a fraction q of the maximum number of keys that the algorithm can track (which we will denote by M to be consisted with the FSS algorithm), then the last element from List1 is inserted into List0.

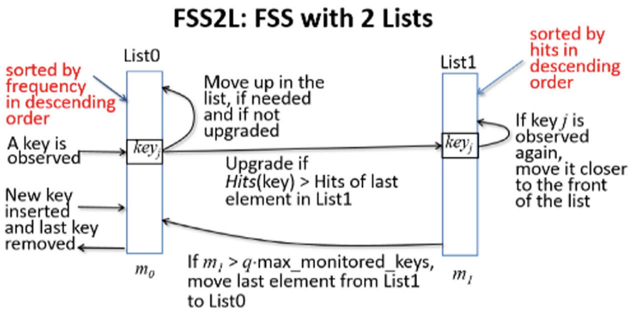

FSS2L: FSS with 2 Lists
List0
List1
sorted by frequency in descending order
sorted by hits in descending order
A key is observed
key_j
key_j
Move up in the list, if needed and if not upgraded
Upgrade if *Hits*(key) > Hits of last element in List1
If key j is observed again, move it closer to the front of the list
New key inserted and last key removed
m_0
m_1
If $m_1 > q$·max_monitored_keys, move last element from List1 to List0

Fig. 2. The keys in List1 are protected in the FSS2L framework from being washed out by the inflow of new keys that are unlikely to be observed again.

If the observed key i is not found in List0 or List1, its key is hashed into the array $H[k]$ and the same procedure as in FSS is followed in order to decide whether or not this key should be inserted into List0 (ahead of all items with the same frequency, so as to stay longer in the list). This procedure implies that if a new key i is inserted into List0, it is assigned estimated frequency $f_i = H[k] + 1$ and a maximum error $\Delta_i = H[k]$. Therefore, its "hits" value (the number of times it was observed after the algorithm started monitoring it) can always be computed dynamically as $h_i = f_i - \Delta_i - 1$.

3.2 FSSA Algorithm

A straight-forward implementation of FSS2L would use two Stream Summary lists of the type described in Sect. 2.3 (for storing information about keys in List0 and List1 and efficiently moving the keys inside these lists or between them) and two hash maps (HashMap0 and HashMap1 with entries indexed by keys from the corresponding lists) for efficiently finding the key of interest in each Stream Summary list (as was suggested at the end of Sect. 2.3).

However, there are cases when it is undesirable to use a linked list implementation. For example, if the data structures need to be written to disk periodically and then read from disk, then the pointer information will become inaccurate. Also, the number of nodes representing keys in List0 can initially grow to M, and then the unused nodes will need to be deallocated as List1 grows to its maximum size of qM while List0 shrinks down to $(1 - q)M$. Deallocation of memory can be hard to implement in some contexts, and the deallocated memory may not be reusable until the algorithm finishes its data processing. These practical concerns motivated us to develop a single array implementation of the FSS2L framework, which we called FSSA algorithm, shown in Fig. 3.

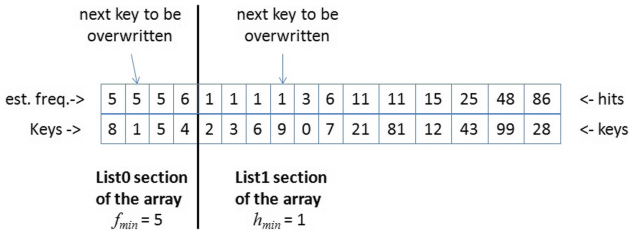


Fig. 3. The 2D array data structure used in FSSA. Keys in the List0 section are sorted by estimated frequency, while those in the List1 section are sorted by observed hits.

This algorithm allocates a 2D array of size M-by-2 to store information about the monitored keys. If the key belongs (conceptually) to List0, then the array stores the key and its estimated frequency. If the key belongs (conceptually) to List1, then the array stores the key and its number of hits. The keys from List0 are kept together at the front of the array and are sorted by frequency, while the keys from List1 are kept together at the end of the array and are sorted by hits.

In order to make insertions of new keys into the List0 section of the array more efficient, instead of always inserting new keys ahead of all other keys with frequency equal to f_{min}, FSSA cycles through those keys and overwrites them one after another. In order to accomplish this, FSSA stores the index of the next key to be overwritten and the largest index for which a key still has frequency equal to f_{min}. Similarly, in order to make “upgrades” of keys from the List0 section to the List1 section more efficient, FSSA stores the index of the next key to be overwritten in List1 and the largest index for which a key still has hits equal to h_{min}. Also, in order to minimize movement of memory within this array, when FSSA increments the estimated frequency for key i and decides to move it closer toward the end of the List0 section, it simply finds, using binary search, the last key that has frequency equal to f_i and swaps it with key i (which will now have frequency $f_i + 1$). A similar strategy is used when incrementing the hits value for a key in the List1 section.

When a new key i is observed and is found in HashMap1, the value of h_i is first computed as $h_i = f_i - \Delta_i - 1$ using f_i and f_i stored in the hash map. Then, the position of key i in the List1 section of the array is determined using binary search for h_i (followed by a short linear search over all keys that have the same hits value). Similarly, if the observed key i is found in HashMap0, then its position in the List0 section of the array is determined using binary search for keys that have a frequency f_i followed by a linear search for the exact key. Therefore, FSSA can perform all of its operations by storing only 5 numbers for each monitored key: the key, its estimated frequency and its maximum error in the hash map and two more numbers in the sorted array. This makes FSSA the most space efficient algorithm out of the ones we considered, as will be discussed in Sect. 4.

3.3 Optimizations that Can Be Added to FSS and FSSA

In order to speed up the operations with a large hash map, the FSS and FSSA can substitute it by an array A of the same size as H, with $A[k]$ storing a hash map for those keys that hash to k. In this way, every operation with the hash map will become faster as it will be done on a much smaller hash map (which is implemented in C++ as a red-black tree with update time being logarithmic in the hash map size). As a result, our experiments showed that the overall execution time for the algorithms was reduced by 25–50% (a larger improvement was observed for lower levels of skew in the frequency distribution of keys).

Another optimization can be added to FSS and FSSA to reduce their space consumption. It is based on the observation made in [5] that with a good uniform hashing of keys into H, the difference between $\max_k H[k]$ and $\max_k H[k]$ is small. Therefore, after observing every n keys (where n can be $10 \cdot sizeH$), $\min_k H[k]$ is computed and is added to a variable H_c that holds the common amount subtracted so far from all cells in H, while all cells in H are decremented by $\min_k H[k]$. As a result, it is sufficient to restrict cells in the hash array to store nonnegative integers that are less than 2^{16} (i.e., take up only 2 bytes each). Then, the criterion for adding to the hash map a new key that hashes to cell k would be $H[k] + H_c \geq f_{min} - 1$. Also, when a key that hashes to cell j is removed from H, we set $H[j] = \max(f_{min} - H_c, H[j])$, where the max() operator is needed because if the removed key was recently moved from List1 to List0, then $f_{min} - H_c$ can be smaller than $H[j]$.

3.4 Adapting the Ratio of List0 Size to List1 Size in FSS2L

During our experiments with different frequency distributions of keys, we discovered that the maximum allowed size of List1, which we have previously denoted by $q \cdot M$ with q being a tunable parameter, has an important effect on the results obtained by the FSS2L framework.

First of all, note that if we let $q = 0$, then FSSA behaves very much like FSS, differing from it only in the ordering of keys within groups that have the same frequency (our experimental results confirm this claim). As was pointed out in Sect. 2.3, the implementation of SS (and by extension of FSS) currently assumed in the literature requires storing 2 numbers in the hash map and 4 numbers in the Stream Summary data structure for each monitored key. However, as was described in Sect. 3.2, FSSA can perform all its operations by storing 3 numbers in the hash map and 2 numbers in the array for each monitored key. Thus, by setting $q = 0$ in the FSSA algorithm, we obtain FSSA(0), which is a more space efficient implementation of FSS than is currently known in the literature.

As q is increased, more and more of the highest frequency keys can enjoy protection on List1 from being displaced by an inflow of infrequently observed keys. However, it also becomes more difficult for keys to get onto List1, since they will spend less time in List0 (whose size is reduced as q is increased). Furthermore, a smaller List0 implies a faster turnover of its keys, which in turn implies a faster growth rate for f_{min}.

When the number of distinct keys in the dataset has the same order of magnitude as the number of entries in the hash array, even a perfectly random hash function will not be able to spread out the keys uniformly over the hash array and some of its cells will have one or zero keys hashed to it. If a medium-frequency key i happens to be the only key hashed to a certain cell $H[k]$ in the array, then we often found that f_{min} grows faster than $H[k]$ if q is set to 0.75 or larger, and key i does not make it into List0. As our experiments in Sect. 5 show, when the number of distinct keys in the data set is small, FSSA(0.75) will miss some of the top n most frequent items, while FSSA(0.25) will discover many of them but will have a larger frequency estimation error than FSSA (0.75). Table 5 shows that the effect described above becomes more noticeable when the degree of skew in the frequency distribution of keys decreases, because it becomes harder for the n th most frequent key to be observed sufficiently many times to catch up with the growing f_{min}.

The above observations suggest that if one is most concerned with minimizing the frequency estimation error, then one should use FSS2L with $q = 0.75$. However, if one is most concerned with not missing any of the top n most frequent items for a value of n that is as large as possible, then one should start with $q = 0.25$, then estimate the number of distinct keys in the dataset, and if this number is found to be much larger than the size of the hash array, then one should increase q to 0.75.

In order to not use too much extra space by running a separate algorithm for estimating the number of distinct keys (such as HyperLogLog described in [4]), it is possible to add a small data structure to the FSS2L framework, which can then be used to determine whether or not the number of keys in the data set is much larger than *sizeH*. This data structure consists of a boolean array B of size *sizeH*-by-9 which is initialized with FALSE values. Whenever a key is observed, hashed to cell k in H and is then not inserted into HashMap0, then if $B[k, 8]$ is FALSE, this key is hashed uniformly into the set {0,1,…,7}, say to a value j, and then the cell $B[k,j]$ is set to TRUE. At that point, if $B[k,j]$ = TRUE for $j \leq 7$, then $B[k, 8]$ is set to TRUE. Then, when the number of observed keys becomes $32 \cdot sizeH$, the cells $B[k, 8]$ are examined for all k, and if the fraction of them with $B[k, 8]$ = TRUE is greater than 0.01, then we can conclude that the number of distinct keys in the dataset is much larger than *sizeH*, and hence the value of q should be increased from 0.25 to 0.75 (if one is most concerned with not missing any of the top n most frequent items). In experiments presented in Sect. 5, this algorithm is called Adaptive FSSA, abbreviated as AFSSA.

4 Space and Time Complexity of Considered Algorithms

All the SS-based algorithms discussed in this paper start with checking whether or not an entry for the observed key exists in the appropriate hash map, which on average takes $O(1)$ time. If the hash map does contain such an entry, then this entry needs to be retrieved, which takes $O(\log(m))$ time. When an entry needs to be removed from the hash map, the $O(\log(m))$ cost is incurred once again. If the modified Stream Summary data structure described in Sect. 2.3 is used to store information about each key and a hash map is used to store pointers into Stream Summary, then all Stream Summary operations take constant time.

Such an implementation requires storing pairs (key, pointer) in the hash structure as well as the key, the pointer to the next node, the pointer to the previous node, and the pointer to the parent bucket in the Stream Summary. Each bucket in the Stream Summary needs to store its frequency as well as the pointers to the first and to the last node in its linked list. In modern database systems each number takes 8 bytes and each pointer uses 8 bytes in order to handle 64-bit memory spaces. Thus, in order to track m keys, SS requires 48 $m + 3b$ bytes, where b is the number of buckets in the Stream Summary. If the frequency distribution of keys has a low skew, then b is likely to be small (usually 2 or 3 in our experiments), but if the skew is large, then b can be close to m.

The authors of the FSS algorithm suggest in [5] that FSS can track only half as many keys as SS and still be more accurate than SS if the same amount of memory is given to both algorithms. They also suggest that the FSS hash array H should have 3 times as many cells as the number of monitored keys (so if FSS monitors $\frac{m}{2}$ keys, then $sizeH = \frac{3m}{2}$). Using these settings and the same data structures as the ones used for SS, all the data structures in FSS besides the hash array will use $24m + 3b$ bytes (assuming, for simplicity, that the same number of buckets b is present in the Stream Summary for both algorithms). If the hash array is implemented using 2 bytes for each cell as was described at the end of Sect. 3.3, then the total space taken by FSS would be equal to $24m + 3b + 2\frac{3m}{2} = 27m + 3b$ bytes in order to track $m/2$ keys. The FSS operations are essentially the same as those of SS, but because it monitors $\frac{m}{2}$ keys, the hash map operations take a little less time than in the SS algorithm, even though they still take on the order of $O(\log(m))$ time. More importantly, however, is that FSS needs to do fewer such operations, since an observed key is not always added to the hash map. Thus, we would expect the FSS algorithm to have a noticeably smaller computation time than SS, which is confirmed by the experiments presented in Sect. 5.

The FSSA algorithm stores 5 numbers for each monitored key, and if its hash array is implemented using 2 bytes for each cell, then its total space consumption would be $20m + 3m = 23m$ bytes if it were to monitor $\frac{m}{2}$ keys. The maximum operational time for FSSA is $O(\log(m))$ because keys in the array are found using binary search.

If one's primary objective is not to miss any of the top n most frequent keys and one decides to add an additional data structure to FSS2L as described in Sect. 3.4, then the space consumption will increase by $\frac{3}{2}m \cdot \frac{9}{8} = \frac{27}{16}m$ bytes. We called the resulting algorithm Adaptive FSSA (AFSSA). Table 1 summarizes the space requirements for the algorithms evaluated in Sect. 5.

5 Experimental Results

5.1 Experiments on Artificial Data

We first evaluated SS, FSS, FSSA and AFSSA algorithms on artificial data sets with keys following Zipf and Exponential distributions. The Zipf(α) distribution was defined over 1 million keys, and then 1 million keys were randomly drawn from this distribution and processed by each algorithm (the nth most frequent key was drawn with probability $P(n) = K/n^{\alpha}$, where K is a constant that makes all probabilities add up to 1). The values of the skew parameter α were chosen to lie between 0.4 and 1.2, which covers the region of interest where detection of frequent items is possible (some skew is present) but is not super easy (the skew is not very large). The number of distinct keys observed in a data set of 10^6 sampled keys ranged, correspondingly, between 600000 and 60000, approximately.

For the Exponential distribution, the probability of observing the n th most frequent key was given by $P(n) = a \cdot e^{-c \cdot n}$, where c is the skew parameter and a is a suitably chosen constant that makes all probabilities add up to 1 on the support [0, NumKeys]. For consistency with experiments performed with the Zipf distribution, NumKeys was chosen to be 1 million. The values of the skew parameter c were chosen to cover the range where detection of frequent items is possible but is not super easy, which corresponded to the range between 0.0003 and 0.0009. The number of distinct keys observed in a data set of 10^6 sampled keys ranged correspondingly, between 21000 and 8000, approximately. Therefore, when evaluating the frequent item detection algorithms for the Exponential distribution with c between 0.0003 and 0.0009, the SS list size m should be much smaller than 8000 in order to make the problem interesting. We chose $m = 1000$ for the experiments described below. In order to make a fair comparison between the algorithms, we set their space consumption (by varying the number of keys each of them is tracking) equal to that of SS with $m = 1000$. The hash array size for FSS and FSSA was set to be 3 times the number of keys each of them monitors, following the suggestion made by the authors of FSS in [5].

The accuracy of the algorithms was measured in several ways. One of them was the number of sequential true top n most frequent keys detected by the algorithm, which we called "topN." For example, if key 1 is the most frequent, key 2 is the next most frequent, and so on, and if the algorithm detects keys 1, 2, 3, 4, 6, … (i.e., it misses key 5), then topN equals 4. However, even if an algorithm detects the top 4 most frequent keys, it can still have very large frequency estimation errors for them. Therefore, in order to record this aspect of accuracy, we also computed the Mean Absolute Error for each algorithm for the top 750 keys (sorted by estimated frequency) on its list.

Tables 1, 2 and 3 show the performance metrics described above, averaged over 50 trials (each cell shows estimated mean and standard error), for the algorithms discussed in this paper under the Zipf distribution of key frequencies. The value of the q parameter for the FSSA algorithm is given in parenthesis in the top row of each table. The last column corresponds to the AFSSA algorithm, where the q parameter was adjusted using the procedure described at the end of Sect. 3.4.

These tables show that FSS is more accurate than SS despite monitoring fewer keys. FSS also runs faster, confirming the theoretical analysis of its run time performed in Sect. 4. As expected, FSSA(0) is more accurate than FSS because both of them have a very similar logic (as was noted at the beginning of Sect. 3.4) but FSSA(0) is more space-efficient than FSS and hence it monitors more keys than FSS.

Tables 1, 2 and 3 also show that for small levels of skew, when many distinct keys are observed, FSSA with $q = 0$ detects a much shorter topN sequence than with $q > 0$. However, for $\alpha = 1.2$ the number of distinct keys observed becomes small enough for the dynamics described in Sect. 3.4 to manifest itself, and consequently FSSA(q) detects the longest topN sequence for small values of q. Despite this fact, FSSA(0.75) has the smallest MAE because whichever keys get into its List1 early on with small estimation errors tend to stay in that list (and hence their errors do not increase). The AFSSA algorithm increases q from 0.25 to 0.75 for $\alpha \leq 1$, but for $\alpha = 1.2$ it detects that the number of observed distinct keys is small enough and does not increase q from its initial value of 0.25. Despite adapting correctly to the shape of the distribution of keys, the performance of AFSSA is hampered by the fact that it uses more space than FSSA and hence is able to track fewer keys. As a result, it is not able to match the best-performing algorithm for each level of skew, but it still detects a longer topN sequence than FSSA(0.25) for the most interesting case of intermediate skew levels.

It is also interesting to observe that FSSA(0.75) and AFSSA (which are implemented using an efficient array implementation) have a smaller run time than FSS (which is implemented using Stream Summary) for $\alpha \leq 0.8$. The reason for this improvement is that whenever a new key needs to be inserted into the array of hash maps (described at the beginning of Sect. 3.3), FSS performs this operation on larger hash maps than FSSA(0.75) or AFSSA, which allocate only 1/4 of the monitored keys to List0.

For large degrees of skew, when $\alpha > 0.8$, insertions of new keys into the hash maps become more and more rare, and the run time of all algorithms decreases. SS algorithm runs much slower than the other algorithms for low degrees of skew because it adds a new key to the hash map whenever it is observed and is not in the hash map, while the other FSS-based algorithms add only those keys that get hashed to a cell in H with a high enough value.

Table 1. Length of the correctly detected topN sequence when keys are drawn from a Zipf(α) distribution

α	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
0.4	**μ = 0.1** s.e. = 0.0	**μ = 0.3** s.e. = 0.1	**μ = 0.5** s.e. = 0.1	**μ = 19** s.e. = 0.9	**μ = 19** s.e. = 0.8	**μ = 16** s.e. = 0.8	**μ = 16** s.e. = 0.8
0.6	**μ = 3.7** s.e. = 0.2	**μ = 20** s.e. = 0.3	**μ = 26** s.e. = 0.5	**μ = 80** s.e. = 2	**μ = 118** s.e. = 2.7	**μ = 145** s.e. = 3.1	**μ = 133** s.e. = 2.6
0.8	**μ = 33** s.e. = 0.5	**μ = 139** s.e. = 0.8	**μ = 168** s.e. = 1	**μ = 197** s.e. = 2	**μ = 250** s.e. = 3	**μ = 299** s.e. = 4	**μ = 277** s.e. = 4
1	**μ = 109** s.e. = 0.9	**μ = 359** s.e. = 1	**μ = 424** s.e. = 2	**μ = 424** s.e. = 1	**μ = 451** s.e. = 2	**μ = 471** s.e. = 3	**μ = 458** s.e. = 3
1.2	**μ = 209** s.e. = 1	**μ = 554** s.e. = 1	**μ = 636** s.e. = 1	**μ = 636** s.e. = 1	**μ = 636** s.e. = 1	**μ = 606** s.e. = 2	**μ = 602** s.e. = 1

Table 2. Mean Absolute Error (MAE) when keys are drawn from a Zipf(α) distribution

α	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
0.4	**μ = 997** s.e. = 0.0	**μ = 291** s.e. = 0.0	**μ = 250** s.e. = 0.0	**μ = 199** s.e. = 0.2	**μ = 160** s.e. = 0.3	**μ = 101** s.e. = 0.3	**μ = 109** s.e. = 0.3
0.6	**μ = 984** s.e. = 0.2	**μ = 275** s.e. = 0.2	**μ = 233** s.e. = 0.1	**μ = 171** s.e. = 0.1	**μ = 101** s.e. = 0.1	**μ = 44** s.e. = 0.2	**μ = 52** s.e. = 0.3
0.8	**μ = 870** s.e. = 0.6	**μ = 190** s.e. = 0.4	**μ = 151** s.e. = 0.3	**μ = 135** s.e. = 0.4	**μ = 72** s.e. = 0.2	**μ = 10** s.e. = 0.	**μ = 17** s.e. = 0.1
1	**μ = 535** s.e. = 0.9	**μ = 70** s.e. = 0.3	**μ = 45** s.e. = 0.4	**μ = 44** s.e. = 0.4	**μ = 34** s.e. = 0.3	**μ = 2.8** s.e. = 0.1	**μ = 6.0** s.e. = 0.1
1.2	**μ = 192** s.e. = 0.5	**μ = 12** s.e. = 0.1	**μ = 3.8** s.e. = 0.0	**μ = 3.8** s.e. = 0.0	**μ = 3.8** s.e. = 0.0	**μ = 1.0** s.e. = 0.0	**μ = 6.7** s.e. = 0.1

Table 3. Run time in ms when keys are drawn from a Zipf(α) distribution

α	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
0.4	**μ = 1313** s.e. = 6	**μ = 474** s.e. = 2	**μ = 535** s.e. = 6	**μ = 488** s.e. = 2	**μ = 440** s.e. = 4	**μ = 432** s.e. = 3	**μ = 420** s.e. = 2
0.6	**μ = 1283** s.e. = 6	**μ = 446** s.e. = 1	**μ = 503** s.e. = 2	**μ = 468** s.e. = 2	**μ = 418** s.e. = 1	**μ = 343** s.e. = 1	**μ = 328** s.e. = 1
0.8	**μ = 1141** s.e. = 4	**μ = 386** s.e. = 3	**μ = 436** s.e. = 2	**μ = 436** s.e. = 2	**μ = 390** s.e. = 2	**μ = 334** s.e. = 2	**μ = 330** s.e. = 4
1	**μ = 777** s.e. = 3	**μ = 244** s.e. = 1	**μ = 356** s.e. = 2	**μ = 343** s.e. = 2	**μ = 337** s.e. = 2	**μ = 315** s.e. = 2	**μ = 304** s.e. = 2
1.2	**μ = 365** s.e. = 1	**μ = 133** s.e. = 1	**μ = 257** s.e. = 1	**μ = 250** s.e. = 2	**μ = 246** s.e. = 2	**μ = 236** s.e. = 2	**μ = 238** s.e. = 1

Tables 4, 5 and 6 show how the considered algorithms perform when the frequency of each key is drawn from the Exponential distribution described at the beginning of this section. Table 4 shows that for a small degree of skew, FSSA(0.25) detects the longest topN sequence, which once again matches the analysis performed in Sect. 3.4.

Table 4. Length of the correctly detected topN sequence when keys are drawn from the Exponential distribution

c	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
0.3	**μ = 0.5** s.e. = 0.1	**μ = 2.2** s.e. = 0.3	**μ = 6.7** s.e. = 0.8	**μ = 11.5** s.e. = 1.3	**μ = 9.2** s.e. = 1	**μ = 4.8** s.e. = 0.6	**μ = 5.2** s.e. = 0.4
0.5	**μ = 0.7** s.e. = 0.2	**μ = 270** s.e. = 2	**μ = 501** s.e. = 2	**μ = 501** s.e. = 2	**μ = 493** s.e. = 2	**μ = 209** s.e. = 8	**μ = 395** s.e. = 2
0.7	**μ = 1.8** s.e. = 0.3	**μ = 512** s.e. = 1	**μ = 715** s.e. = 1	**μ = 716** s.e. = 1	**μ = 716** s.e. = 1	**μ = 549** s.e. = 5	**μ = 660** s.e. = 1
0.9	**μ = 10** s.e. = 1	**μ = 676** s.e. = 1	**μ = 744** s.e. = 1	**μ = 744** s.e. = 1	**μ = 744** s.e. = 1	**μ = 669** s.e. = 3	**μ = 736** s.e. = 1

Table 5. Mean Absolute Error (MAE) when keys are drawn from the Exponential distribution

c	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
0.3	**μ = 822** s.e. = 0.4	**μ = 190** s.e. = 0.5	**μ = 129** s.e. = 0.9	**μ = 97** s.e. = 0.5	**μ = 52** s.e. = 0.4	**μ = 2.9** s.e. = 0.1	**μ = 119** s.e. = 0.3
0.5	**μ = 716** s.e. = 0.7	**μ = 79** s.e. = 0.4	**μ = 23** s.e. = 0.2	**μ = 23** s.e. = 0.2	**μ = 18** s.e. = 0.1	**μ = 2.0** s.e. = 0.1	**μ = 43** s.e. = 0.3
0.7	**μ = 626** s.e. = 0.9	**μ = 24** s.e. = 0.2	**μ = 2.15** s.e. = 0.0	**μ = 2.15** s.e. = 0.0	**μ = 2.15** s.e. = 0.0	**μ = 1** s.e. = 0.1	**μ = 6.55** s.e. = 0.1
0.9	**μ = 481** s.e. = 2	**μ = 6.5** s.e. = 0.1	**μ = 0.54** s.e. = 0.0	**μ = 0.54** s.e. = 0.0	**μ = 0.54** s.e. = 0.0	**μ = 0.48** s.e. = 0.0	**μ = 1.0** s.e. = 0.0

Table 6. Run time in ms when keys are drawn from the Exponential distribution

c	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
0.3	**μ = 1061** s.e. = 7	**μ = 328** s.e. = 2	**μ = 402** s.e. = 2	**μ = 437** s.e. = 2	**μ = 409** s.e. = 2	**μ = 371** s.e. = 3	**μ = 413** s.e. = 6
0.5	**μ = 952** s.e. = 3	**μ = 260** s.e. = 2	**μ = 360** s.e. = 2	**μ = 407** s.e. = 2	**μ = 394** s.e. = 2	**μ = 387** s.e. = 2	**μ = 401** s.e. = 3
0.7	**μ = 826** s.e. = 3	**μ = 225** s.e. = 1	**μ = 346** s.e. = 2	**μ = 398** s.e. = 2	**μ = 402** s.e. = 3	**μ = 389** s.e. = 2	**μ = 378** s.e. = 2
0.9	**μ = 748** s.e. = 4	**μ = 209** s.e. = 1	**μ = 347** s.e. = 2	**μ = 405** s.e. = 3	**μ = 397** s.e. = 2	**μ = 393** s.e. = 2	**μ = 372** s.e. = 2

When keys are sampled from the Exponential distribution, the number of distinct keys observed is not much greater than the size of the hash array H, which makes it difficult for medium-frequency keys to get into List0. Furthermore, when the degree of skew in the distribution of key frequencies decreases, it becomes more difficult for the nth most frequent key to differentiate itself from the less frequent keys, which exacerbates the dynamics described in Sect. 3.4 and causes FSSA(0.25) to detect longer topN sequences than FSSA(q) for $q > 0.25$ when $c = 0.0003$.

The AFSSA algorithm does not increase the q value from its initial setting of 0.25 because it detects that the data set does not have too many distinct keys and. As a result, AFSSA detects a longer topN sequence than FSSA(0.75). Comparing this with the previous experiments on the Zipf distribution, we see that AFSSA provides a robust mid-level performance and is able to avoid, through adaptation, the poor performance sometimes suffered by each fixed FSSA algorithm.

Another interesting observation is that the run time of AFSSA is smaller than that of FSS for a small degree of skew in the Zipf distribution, while for the Exponential distribution it is always larger than that of FSS. This happens because the Exponential distribution has a much smaller number of distinct keys and hence new keys are inserted into the hash map relatively less frequently, thus reducing the disadvantage of FSS due to its larger hash maps for List0. On the other hand, updates of existing keys happen relatively more frequently, and such updates require a binary search for the key in List1 of AFSSA, while FSS contains pointers in the List1 hash maps to each key in its Stream Summary.

5.2 Experiments on Real Data

We now present experimental results on the data sets chosen from the "Open-Source Data Mining Library" [9]. We selected four largest data sets with the largest number of distinct keys. The first data set was "Kosarak", which contains anonymized click-stream data from a Hungarian news portal (8019015 keys, 41270 of which are distinct). The second data set was "Retail", which contains anonymized retail market basket data from an anonymous Belgian retail store (908576 keys, 16470 of which are distinct). The third data set was "BMS2", which contains anonymized click-stream and purchase data from Gazelle.com, a legwear and legcare web retailer that closed their online store on 8/18/2000. This data set was used in the KDD-Cup 2000 competition and it contains 358278 keys, 3340 of which are distinct. The fourth data set was "Bible", where each key corresponds to a distinct word in the Bible (787066 keys, 13905 distinct).

Performance of the considered algorithms on these data sets is shown in Tables 7, 8 and 9. Since the number of distinct keys in these data sets covered the range similar to that seen in the Exponential distribution, the observed results follow the same pattern as the one observed in Tables 4, 5 and 6.

Table 7. Length of the correctly detected topN sequence for real data sets

	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
Kosarak	178	567	743	743	743	722	711
Retail	114	495	618	618	618	357	581
BMS2	337	700	728	729	728	528	722
Bible	359	682	703	703	703	633	694

Table 8. Mean Absolute Error (MAE) for real data sets

	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
Kosarak	2675	67	1.8	1.8	1.8	1.0	8.9
Retail	435	40	19	19	19	10	24
BMS2	82	6	3.3	3.2	3.2	2.5	3.9
Bible	94	5.3	3.0	3.0	3.0	2.0	3.6

Table 9. Run time in ms for real data sets

	SS	FSS	FSSA(0)	FSSA(0.25)	FSSA(0.5)	FSSA(0.75)	AFSSA
Kosarak	4424	1198	2153	2333	2356	2312	2316
Retail	589	167	287	330	322	299	306
BMS2	164	63	139	130	159	144	136
Bible	195	80	224	215	201	192	198

6 Conclusion

This paper presented a new 2-list framework (called FSS2L) for detecting the most frequent keys in a data set and for accurately estimating their frequencies. We also presented a space-efficient implementation of this framework using a single array FSSA(q) algorithm, where the parameter q controls the relative sizes of the two lists. We showed that FSSA(0) performs the same operations as the previously published FSS algorithm [5] but using less space than the "standard" implementation assumed in the literature (e.g., [1, 2]) based on the Stream-Summary data structure. As a result, FSSA(0) gives more accurate results than FSS when their space consumption is equalized.

Dependence of FSSA on the parameter q was studied analytically and experimentally. Our results showed that if one is primarily interested in detecting most accurately the top n most frequent items, then one should use a small value of q (0.25 was shown to work well in our experiments) if the number of distinct keys in the data set is not much larger than the size of the hash array H used by FSS2L; otherwise, one should use a large value of q (0.75 was shown to work well in our experiments).

Accurate detection of the top n most frequent keys is the important accuracy measure when a distributed (or parallel) hash join needs to be performed by a database engine. In this case, in order to minimize the overall execution time, no single node (or process) should be overwhelmed by needing to process a join with a very frequent key. In order to achieve this goal, the most frequent items are detected by analyzing a sample of the data and then information about these items is broadcast to all nodes (or processes). The execution time of this scheme depends on the frequency of the first true most frequent item that is not detected by the algorithm, since processing of this item by a single node (or process) to which this item is hashed will be the bottleneck for the overall distributed join scheme. Therefore, the long sequence of the true top n consecutive most frequent items detected by the FSSA algorithm makes it an especially attractive algorithm for use with distributed hash joins.

We have also described an adaptive version of FSSA (called AFSSA), which starts with $q = 0.25$, then estimates the number of distinct keys in the data set, and then increases q to 0.75 if the number of distinct keys was found to be much larger than the size of H. Our experiments on both artificial and real data sets showed that AFSSA is able to adapt to the shape of the distribution of keys and avoid the poor performance sometimes suffered by fixed FSSA(0.25) and FSSA(0.75) algorithms. However, the smallest mean absolute error (MAE) in frequency estimates was consistently obtained by FSSA(0.75), and so if one is primarily interested in minimizing MAE while detecting keys that are reasonably frequent, then one should use FSSA with $q = 0.75$.

FSS2L algorithms can be naturally parallelized for use in a multi-process environment. This can be achieved by breaking up a large data set into P equal parts and then assigning each part i to be scanned by process p_{1i}. While scanning, each process p_{1i} hashes scanned keys into the range [0, $P \cdot R$], and then the keys that hash into [0,R] get sent to process p_{21}, those that hash into $[R, 2R]$ get sent to process p_{22}, etc. Each process p_{2j} then uses the same hash function to hash the incoming keys into its hash array of size R. As a result, the processes p_{2j} would work with disjoint sets of keys, and assuming that each of them uses as much memory as in the serial case, the accuracy guarantee $0 \leq \Delta_i \leq H[k] \leq f_{min} \leq 2N/(m \cdot P)$ would apply for the estimated frequency of each key i, with the factor P appearing in the denominator because each process p_{2j} is expected to observe only N/P keys. Then, at the end of the data set, a "*top n*" query would take the top n most frequent elements from each process, merge together the resulting P top n lists, and then return top n elements from the merged list. This parallelized version of FSSA has been integrated into the Oracle database.

References

1. Cormode, G., Hadjieleftheriou, M.: Finding frequent items in data streams. VLDB Endowment **1**(2), 1530–1541 (2008)
2. Das, S., Antony, S., Agrawal, D., El Abbadi, A.: Thread cooperation in multicore architectures for frequency counting over multiple data streams. VLDB Endowment **2**(1), 217–228 (2009)
3. Demaine, E., López-Ortiz A., Munro, J.I.: Frequency estimation of internet packet streams with limited space. In: Proceedings of the European Symposium on Algorithms (ESA), pp. 348–360 (2002)
4. Flajolet, P., Fusy, E., Gandouet, O., Meunier, F.: Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In: Proceedings of the 13th Conference on Analysis of Algorithm, pp. 127–146 (2007)
5. Homem, N., Carvalho, J.: Finding top-k elements in data streams. Inf. Sci. **180**(24), 4958–4974 (2010)
6. Manku, G., Motwani R.: Approximate frequency counts over data streams. In: Proceedings of 28th International Conference on Very Large Data Bases (VLDB), pp. 346–357. Morgan Kaufmann, Hong Kong (2002)
7. Manerikar, N., Palpanas, T.: Frequent items in streaming data: an experimental evaluation of the state-of-the-art. Data Knowl. Eng. **68**(4), 415–430 (2009)
8. Metwally, A., Agrawal, D., El Abbadi, A.: Efficient computation of frequent and top-k elements in data streams. In: Eiter, T., Libkin, L. (eds.) ICDT 2005. LNCS, vol. 3363, pp. 398–412. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30570-5_27
9. Open-Source Data Mining Library. http://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php

Evaluating Frequent-Set Mining Approaches in Machine-Learning Problems with Several Attributes: A Case Study in Healthcare

Shruti Kaushik[1](✉), Abhinav Choudhury[1](✉), Nataraj Dasgupta[2], Sayee Natarajan[2], Larry A. Pickett[2], and Varun Dutt[1](✉)

[1] Applied Cognitive Science Laboratory, Indian Institute of Technology Mandi, Mandi 175005, Himachal Pradesh, India
{shruti_kaushik, abhinav_choudhury}@students.iitmandi.ac.in, varun@iitmandi.ac.in

[2] RxDataScience, Inc., New York 27709, USA
{nd, sayee, larry}@rxdatascience.com

Abstract. Often datasets may involve thousands of attributes, and it is important to discover relevant features for machine-learning (ML) algorithms. Here, approaches that reduce or select features may become difficult to apply, and feature discovery may be made using frequent-set mining approaches. In this paper, we use the Apriori frequent-set mining approach to discover the most frequently occurring features from among thousands of features in datasets where patients consume pain medications. We use these frequently occurring features along with other demographic and clinical features in specific ML algorithms and compare algorithms' accuracies for classifying the type and frequency of consumption of pain medications. Results revealed that Apriori implementation for features discovery improved the performance of a large majority of ML algorithms and decision tree performed better among many ML algorithms. The main implication of our analyses is in helping the machine-learning community solves problems involving thousands of attributes.

Keywords: Apriori algorithm · Frequent-set mining · Machine learning Pain medications · Features

1 Introduction

Since the early 90s, machine-learning (ML) algorithms have been used to help mine patterns in data sets concerning fraud detection and others [1]. In recent years, ML algorithms have also been utilized in the healthcare sector [2]. In fact, the existence of electronic health records (EHRs) has allowed researchers to apply ML algorithms to learn hidden patterns in data to improve patient outcomes like the type of medications patients consume and the frequency at which they consume these medications [3]. Mining hidden patterns in healthcare data sets could help healthcare providers and pharmaceutical companies to plan quality healthcare for patients in need.

To predict healthcare outcomes accurately, ML algorithms need to focus on discovering appropriate features from data [4]. In general, healthcare data sets are large, and they may contain several thousands of attributes to enable learning of patterns in data [5]. The presence of many attributes in data sets may make it difficult to discover the most relevant features for predicting outcomes via ML algorithms.

Presence of thousands of attributes in the data is problematic for classification algorithms for processing these attributes as features require large memory usage and high computational costs [37]. Two techniques have been suggested in the literature to address the problem of datasets possessing a large number of features: feature reduction (dimensionality reduction) and feature selection [38]. Feature reduction technique reduces the number of attributes by creating new combinations of attributes; whereas, feature selection techniques include and exclude attributes present in the data without changing them [38]. Popular algorithms like Principal Component Analysis (PCA; for features reduction) and Analysis of variance (ANOVA; for features selection) have been used for datasets with a large number of features in the past [6, 7]. PCA is a linear feature-based approach that uses eigenvector analysis to determine critical variables in a high dimensional data without much loss of information [8]. ANOVA is a collection of statistical models used to analyze the differences between group means and their associated procedures (such as "variation" among and between groups) [9]. In ANOVA, the features that describe the most substantial proportion of the variance are the features that are retained in data [10]. Although both PCA and ANOVA approach seem to help in feature discovery, these approaches may become computationally expensive to apply in problems where there are thousands of features in data (e.g., thousands of diagnostic and procedure codes across several patient cases in medical datasets). Another disadvantage of the PCA method is that it is an elimination technique that considers a single feature to be important or unimportant to the problem rather than a group of features being important [8]. Similarly, in ANOVA, researchers need to test assumptions of normality and independence, which may not be the case when features depend upon each other [9, 10]. One way to address the challenge posed by data sets with several thousands of features is by using frequent item-set mining algorithms (e.g., Apriori algorithm) to discover a subset of features because these algorithms look at the associations among items while selecting frequent item-sets [11]. The primary goal of this paper is to evaluate the potential of Apriori frequent item-set mining algorithm for feature discovery before application of different ML algorithms. Specifically, we take a healthcare dataset involving consumption of two pain medications in the US, and we apply different ML algorithms both with and without a prior feature-discovery process involving the Apriori algorithm. The Apriori algorithm works on the fundamental property that an item-set is frequent only if all its non-empty subsets are also frequent [11]. Using the Apriori algorithm, we generate frequently appearing diagnosis and procedure codes in a healthcare dataset. Then, using these frequently occurring diagnosis and procedure codes as present/absent features, along with other features, we apply certain supervised ML algorithms. We check the benefits of using the Apriori algorithm by comparing the classification accuracies of certain ML algorithms when all attributes are considered as features in the dataset and when only the discovered attributes via Apriori are considered as features. To get confidence in our results, we replicate our analyses using several ML algorithms such as the decision tree [27], Naïve Bayes classifier [30, 41], logistic regression [31], and support vector machine [33, 42].

In what follows, we first provide a brief review of related literature. In Sect. 3, we explain the methodology of using the Apriori algorithm for features discovery. In Sect. 4, we present our experimental results and compare classification accuracies for cases with or without the Apriori implementation. We conclude our paper and provide a brief discussion on the implication of this research and its future scope.

2 Background

Data analytics could help in providing useful insights in very large healthcare datasets [12]. These insights may aid in effective decision-making and save lives [12]. Researchers have applied many such techniques to mine the hidden knowledge in the medical domain [15]. For example, various data-mining approaches such as classification, clustering, statistical approaches, and association-rule mining have made a significant contribution to healthcare research [13, 15].

Healthcare researchers have used different ML algorithms to investigate research questions in healthcare [20–22]. For example, some researchers have used the Naive Bayes classification algorithm to diagnose heart diseases [20]. Others have used ML techniques like J48, MLP, Random Forest, SVM, and Bayesian Network classifiers to classify liver-disease patients [21]. Researchers have also used frequent-set mining approaches in the recent past [14–18]. For example, in the literature on frequent-set mining, some researchers have introduced the frequent-set mining pincer-search algorithm to discover the maximum frequent-item sets [15, 17]. Similarly, Rani, Prakash, and Govardhan [16] presented a model for multilevel association rule mining, which satisfies the different minimum support at each level. Also, certain researchers have focused on identifying frequent diseases using the frequent-set mining algorithms like Apriori [14] and mined different association rules for consumers of certain pain medications [19].

Furthermore, prior research has combined methods like PCA with ML algorithms for feature extraction and subsequent classification. For example, certain researchers have used traditional methods like PCA, rough PCA, unsupervised quick reduction algorithm, and empirical distribution ranking approach to extract features that could be further used for an ML classification task [22]. However, to the best of authors' knowledge, prior research has yet to combine frequent-set mining algorithms along with ML algorithms in feature-discovery and predicting healthcare outcomes. In this paper, we attend to this literature gap and apply the Apriori frequent-set mining algorithm for features discovery before performing machine learning for classifying healthcare outcomes. To test our Apriori approach, we take certain healthcare dataset where there are potentially thousands of features to choose between and where feature selection via traditional methods may become computationally expensive. We investigate the performance of different ML algorithms with and without frequent-set mining using the Apriori approach.

3 Method

3.1 Data

In this paper, we use the Truven MarketScan® health dataset containing patients' insurance claims in the US [23]. The data set contains 120,000 patients, who consumed two pain medications, medicine A, medicine B, or both between January 2011 and December 2015.[1] The dataset contains patients' demographic variables (age, gender, region, and birth year), clinical variables (admission type, diagnoses made, and procedures performed), the name of medicines, and medicines' refill counts per patient. We used a big-data architecture consisting of q-programming language to query a kdb+ database [24] for fetching patient records who have consumed pain medication A, B, or both during their journeys. After fetching data from the kdb+ database, we prepared a file containing different diagnoses and procedures corresponding to different patients, who consumed both medications. The dataset contains 55.20% records of patients who consumed medicine A only, 39.98% records of medicine B only, and 4.82% records for those patients who consumed both these medications. There were 15,081 attributes present in total against each patient in this dataset. These attributes consist of patients' age, gender, region, type of admission, diagnoses and procedures performed on the patient, medicine name and its refill information. Out of 15,081 attributes, 15,075 attributes were diagnoses and procedure codes some of which were inter-related.

The diagnoses and procedures were written for patients using the International Classification of Diseases (ICD)-9 codes [25]. The ICD codes are used by physicians and other healthcare providers to classify different diagnoses and procedures recorded during different illnesses in the United States [25]. We applied the Apriori frequent-set mining algorithm to diagnoses made and procedures performed for different patients consuming the two pain medications. The Apriori algorithm discovered the frequently appearing diagnoses and procedures among the 15,075 unique diagnoses procedure codes available in the dataset. We used these frequently occurring diagnoses and procedures as input features along with other independent variables in different ML algorithms that were applied to our dataset. The ML algorithms classified patients according to the type of medications consumed and the frequency of refilling different medications in the dataset.

3.2 Association-Rule Mining

Association-rule mining [36] is a popular technique that aims to extract associations among items in data. An association rule represents a relationship between a group of objects in the database. The basic model of association-rule mining is described below.

Let $I = \{I_1, I_2, \ldots, I_n\}$ be a set of n distinct items, where each attribute $I_1, I_2 \ldots I_n$ is binary (0 or 1) in nature. T is a transaction with a unique transaction id that contains a subset of items in I. Let D be a database having different transaction records, where each transaction is differentiated by the transaction ID and may contain a subset of

[1] Due to a non-disclosure agreement, we have anonymized the actual names of these medications.

items in I. Thus, D = {T_1, T_2, …, T_M}. An association rule is an implication in the form of X → Y, where X, Y ⊆ I and X ∩ Y = ∅. X is called an antecedent while Y is called consequent. There are two measures for finding association rules: support (S) and confidence (C). Support of an item-set X is defined as the proportion of the transactions that contain the item-set X in the database D.

$$Support(X) = \frac{count(X)}{count(D)} \tag{1}$$

The confidence of an association rule X → Y is defined as the proportion of transactions that contain both items X and Y in all the transactions that contain item X.

$$Confidence(X \rightarrow Y) = P(Y \mid X) = \frac{Support(X \cup Y)}{Support(X)} \tag{2}$$

The confidence is a measure of the strength of the association rules. If there is an association rule "X → Y" whose support and confidence satisfies minimum support threshold (min_support) and minimum confidence threshold (min_conf) provided by a user, then we call it an association rule with min_support and min_conf.

3.2.1 Apriori Algorithm

The Apriori algorithm [11] finds frequent item-sets using an iterative level-wise approach based on candidate generation. This algorithm works in following steps:

1. The transactions in database D are scanned to determine frequent 1-itemsets, L_1 that possess the minimum support.
2. Generate candidate k item-sets C_k from joining two k − 1 itemsets, L_{k-1}, and remove its infrequent subset.
3. Scan D to get support count for each k item-sets, C_k.
4. The set of frequent k item-sets, L_k, is then determined. L_k results from support count of candidate k − 1 item-sets.
5. Back to step 2 until there is no candidate k + 1 item-sets, C_{k+1}.
6. Extract the frequent k item-sets, L = L_k.

The Apriori algorithm was used to mine the frequent diagnoses and procedures out of 15,075 unique diagnoses and procedures present in the dataset. After getting the 9 frequent diagnoses and procedures from the result of Apriori algorithm, we formed two ML problems. In the first problem, we used the frequent diagnoses and procedure codes as categorical variables along with the other 6-independent variables to classify patients by the medication they consumed; i.e., consuming medicine A, B, or both. Thus, the first ML problem is a three-class problem. In the second ML problem, we used the frequent diagnoses and procedure codes obtained from the result of Apriori algorithm along with the other 6-independent variables to classify patients as frequent or infrequent buyers of medications they consumed. This ML problem is a two-class classification problem.

We then applied different ML algorithms to all 15,081 attributes in the dataset and compared the classification results with the case when only 15 attributes were used in the ML algorithms. Next, we discuss the brief descriptions of different ML algorithms that we used in this paper, i.e., decision tree [27], Naïve Bayes [30, 41], logistic regression [31], and support vector machine [33, 42]. We applied ZeroR [26] as a base line algorithm to compare the classification accuracies of different ML algorithms mentioned above.

3.3 Machine-Learning Algorithms

3.3.1 ZeroR

ZeroR is the simplest classification method which relies on the target to be classified and ignores all predictors [26]. The ZeroR classifier simply predicts all points as belonging to the majority class. Although there is no predictability power in ZeroR, it is useful for determining a baseline performance as a benchmark for other classification methods.

3.3.2 Decision Tree

A decision tree is a tree, where non-leaf nodes denote tests on attributes, each branch denotes the result of different tests, and each leaf node denotes the class label [27]. In the decision tree, each internal node is labelled with an input feature, and leaf of the tree either gives a class label or a probability distribution over the classes. The results obtained from decision trees are easier to interpret. Following assumptions are taken into account while creating a decision tree [28]:

1. Initially, the complete training set is considered as the root.
2. Feature values are preferred to be categorical. If the values are continuous, then they are discretized before building the model.
3. Records are distributed recursively by attribute values.
4. Order of placing attributes as root or internal node of the decision tree is done by using a statistical approach based on the calculation of entropy and gain.

The first challenge in a decision tree implementation is to identify which of the attributes to select as the root node and at each level. Random selection of nodes may give bad results with very low accuracy. Handling this problem is known as the attributes selection. We have used the information-gain measure to identify the attribute that can be considered as the root node at each level [39].

3.3.3 Naïve Bayes

This classifier belongs to a family of probabilistic classifiers that is based on the Bayes theorem with strong independence assumptions between features [30, 41]. It assumes that the value of a particular feature is independent of the value of any other feature, given the class variable. This classifier attempts to maximize the posterior probability in determining the class of a transaction. A Naïve Bayes classifier assumes features to contribute independently to the probability, regardless of any correlation between features.

Suppose, n be the number of features in a problem which is represented by a vector y = (y_1, y_2, …,y_n) and K be the possible number of classes C_k. Naïve Bayes is a conditional probability model which can be decomposed as [40]:

$$p(C_k/y) = \frac{p(C_k)p(y/C_k)}{p(y)} \tag{3}$$

In practice, the numerator of this equation determines the LHS (the denominator is a constant). Under the independence assumption among attributes, the probability of certain attributes belonging to a certain class is defined as [40]

$$p(C_k/y_1, \ldots, y_n) = p(C_k)\prod\nolimits_{i=1}^{n} p(y_i/C_k) \tag{4}$$

A common rule is to pick the class that is the most probable. This most probable class is defined by the maximum a posteriori (MAP) decision rule [40] as:

$$y = argmax_{k\in 1\ldots K}\, p(C_k)\prod\nolimits_{i=1}^{n} p(y_i/C_k) \tag{5}$$

3.3.4 Logistic Regression

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval, or ratio-level independent variables [31]. The dependent variable in logistic regression or logit model is categorical. Logistic regression is named for the function used at the core of the method, the logistic function [32]. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits. Input values (x) are combined linearly using weights or coefficient values to predict an output value (y). The simple logistic regression is defined as:

$$y = e^{(b0 + b1*x)/\left(1 + e^{(b0 + b1*x)}\right)} \tag{6}$$

where y is the predicted output, b0 is the bias or intercept term, and b1 is the coefficient for the single input value (x). In the general logistics regression model, each column in data has an associated b coefficient (a constant real value) that must be learned from the training data.

3.3.5 Support Vector Machines

Support vector machines are supervised learning models that are binary classification algorithms [33]. Support vector machines construct a hyperplane or a set of hyperplanes in a high dimensional space that can be used for classification, regression, or other tasks. There are two types of support vector machines: linear SVM and the non-linear SVM [42]. If the data is linearly separable, then the linear SVM is sufficient to perform classification. However, if the problem cannot be classified linearly, then we require a non-linear SVM to perform the task. The non-linear support vector machine function takes the data into the high dimensional plane and then performs classification [42]. In the SVM algorithm, we optimize the support weights to minimize the objective (error) function for better classifications.

3.4 Model Calibration

3.4.1 Apriori Implementation

To find the frequent diagnoses and procedure codes using Apriori algorithm, we needed to set the threshold limits for support and confidence. For setting the minimum threshold limits for support and confidence, we conducted sensitivity analyses. We first calculated the male-to-female ratio among patients who consumed either medication A, B, or both in the dataset. This ratio turned out to be 0.58. Then, we tried different values of minimum support and minimum confidence till the point when the association rules obtained from Apriori algorithm had the same male-to-female ratio of 0.58 as in the dataset. With 3% threshold support and 99% threshold confidence, we got the similar male-to-female ratio for all the association rules with frequent diagnoses and procedure codes. We got three association rules when we applied the Apriori algorithm. These rules were made up of the following nine ICD-9 codes: total knee arthroplasty, osteoarthrosis secondary lower leg, removal of foreign body from the eye, total knee replacement, osteoarthrosis primary lower leg, osteoarthrosis generalized lower leg, total hip arthroplasty, iridectomy, and total hip replacement.

3.4.2 Machine Learning Combined with Apriori

For the ML analyses, the dataset was randomly divided into two parts: 70% of the data was used for training, and 30% of the data was used for testing. We used the d-prime = z (true-positive rate) – z (false-positive rate) as measure of accuracy [34, 43]. The higher the d-prime, the better the performance (a d-prime = 0 indicates random performance, where true-positive rate = false-positive rate). Our first machine learning problem is a three-class problem, where we classified a patient according to the medication consumption. So, a patient can be classified under class A, class B or both. Our second ML problem is a two-class problem, where we classified patients according to their frequency of medicine consumption. So, a patient can be classified as a frequent buyer or an infrequent buyer of medications. We took the median of refill counts per patient (=3) to distinguish between a frequent buyer (>3) and an infrequent buyer (≤ 3). In the dataset, 41.11% patients belong to the frequent class and rest to the infrequent class. We used the frequent codes obtained from Apriori algorithm as categorical variables along with the other demographic and clinical features while training our ML models. For different classification problems, we used different features from the original dataset and the Apriori output (see Table 1). Table 1 shows the list of 15-features used in different ML models after applying Apriori procedure for the three-class and two-class classification problems. As shown in Table 1, some of these 15-features were excluded in certain problems (e.g., refill count was excluded in the

Table 1. Description of input features for classification problems and their source

Features	Description	Features for 3-class problem	Features for 2-class problem
Sex	Male Female	Included	Included
Age group	0–17, 18–34, 35–44, 45–54, 55–64	Included	Included
Region	Northeast, northcentral, south, west, unknown	Included	Included
Type of admission	Surgical, medical, maternity and newborn, psych and substance abuse, unknown	Included	Included
Refill count	Count in number	Included	Excluded
Pain medication	A, B, Both	Excluded	Included
Total knee arthroplasty	Present/not present	Included	Included
Osteoarthrosis of secondary lower leg	Present/not present	Included	Included
Removal of foreign body from eye	Present/not present	Included	Included
Total knee replacement	Present/not present	Included	Included
Osteoarthrosis of primary lower leg	Present/not present	Included	Included
Osteoarthrosis of generalized lower leg	Present/not present	Included	Included
Total hip arthroplasty	Present/not present	Included	Included
Iridectomy	Present/not present	Included	Included
Total hip replacement	Present/not present	Included	Included

two-class problem). That is because these attributes were strongly correlated with the predicted class. Certain features of this table were directly included from the dataset (sex, age group, region, type of admission, refill count and pain medication) while rest were included after applying Apriori on the procedure and diagnoses codes.

For both the classification problems, we ran two different datasets on different ML models. The first dataset contained features obtained from the Apriori algorithm along with other demographic and clinical features from the dataset; whereas, the second dataset contained all the features (=15,081).

4 Results

4.1 Apriori Algorithm

Based on the Apriori algorithm [11], we found the following three association rules among patients who consumed either medicine A, medicine B, or both during their patient journeys:

1. *If a patient goes for* ***total knee arthroplasty****,* ***osteoarthrosis of primary/ secondary/generalized lower leg*** *and* ***removal of foreign body from posterior eye segments****, then he/she goes* ***for total knee replacement*** *and consumes A/B.*
2. *If a patient goes for total knee arthroplasty, then he/she goes for total knee replacement and consumes A/B.*
3. *If a patient goes for* ***total hip arthroplasty*** *and* ***Iridectomy****, then he/she goes for* ***total hip replacement*** *and consumes both the medications.*

As explained above, we took nine frequently occurring ICD-9 diagnoses and procedures codes from the rules above. These codes have been bolded in the rules.

4.2 Machine-Learning Algorithms

We applied various ML algorithms like Naïve Bayes, Decision Tree, Logistic Regression, Support Vector Machine (linear kernel), and Support Vector Machine (radial kernel) [26–33] on our dataset and compared their classification accuracy using d-prime. Figure 1 shows the d-prime results from different ML algorithms for the three-class problems (Fig. 1A) and two-class problems (Fig. 1B) with and without Apriori algorithm implementation. We have only used 1000 features for SVMs as the algorithm was not able to scale for 15,081 features (in without Apriori implementation). In the three-class problem (Fig. 1A), without the Apriori implementation, the best d-prime was obtained by the SVM with a radial kernel function. However, with the Apriori implementation in the three-class problem, the best performance was obtained by the decision tree. The performance of all the algorithms improved with the implementation of Apriori algorithm. Furthermore, in the two-class problem (Fig. 1B), both with and without the Apriori implementation, the best d-prime was obtained by the decision tree. In fact, barring the decision tree, all other algorithms possessed a d-prime = 0 (true-positive rate = false-positive rate) in the two-class problem without the Apriori implementation. Barring the SVM with linear kernel and ZeroR algorithms, the performance of all other algorithms improved with the implementation of Apriori algorithm. In general, for both with and without Apriori implementation, all algorithms performed better (higher d-prime) in the three-class problem compared to the two-class problem.

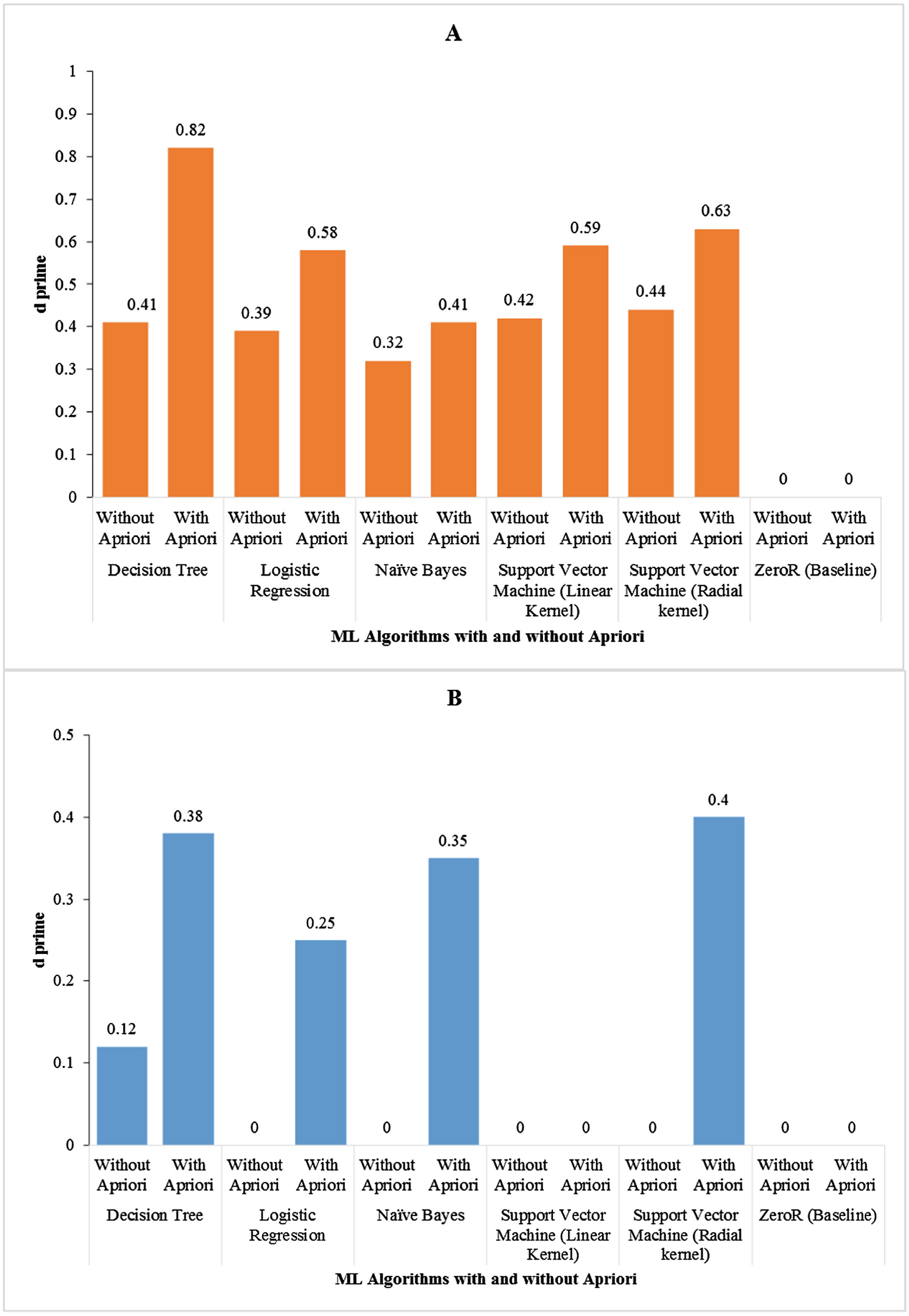


Fig. 1. The d-prime results from different ML algorithms for the three-class problems (A) and two-class problems (B) with and without Apriori algorithm implementation.

5 Discussion and Conclusions

Medical data concerning patient and their journeys may likely contain a large number of attributes detailing demographics as well as procedural and diagnostic information [44, 45]. Thus, before attempting different machine-learning (ML) algorithms on patient-journey datasets, it would be good to reduce the number of attributes used as features. One way to address this reduction is by using frequent item-set mining algorithms (e.g., Apriori algorithm). These algorithms help discover a subset of features by evaluating the associations among items in different transactions [11]. The primary objective of this paper was to evaluate the potential of Apriori frequent item-set mining algorithm for features discovery before application of different ML algorithms. First, using Apriori algorithm, we discovered the frequently occurring attributes (diagnoses and procedures) among patients consuming pain medications. The Apriori frequent-set mining approach gave nine frequently occurring diagnoses and procedure attributes in association rules out of a total of 15,075 possible attributes in the dataset. Second, we found that the Apriori implementation led to improved performance from ML algorithms: in general, the d-prime was higher after application of Apriori compared to when Apriori was not applied, and all attributes were considered in the algorithms. Third, we found that the ML algorithms classified the patients according to the medication used (the three-class problem) better compared to the frequency of medication used (the two-class problem). Finally, we found that the decision-tree algorithm performed better compared to a large number of ML algorithms across both the three-class and two-class problems.

First, we found that using the Apriori implementation [11] improved the performance of a large majority of ML algorithms. One likely reason for this finding is that Apriori procedures allow us to find features that frequently occur together or are correlated with each other. For example, the nine-attributes selected by the Apriori algorithm out of a total of 15,075 attributes occurred in three association rules that possessed the confidence of 99%. Given the high confidence of these rules, the attributes present in them were highly correlated. As the rules predicted the use of medications, these attributes seemed to predict the medications' use and their frequency well. Overall, given our results, the Apriori algorithm seems to be a suitable technique for identifying important features, when the dataset contains thousands of relevant or irrelevant attributes.

Third, we found that the ML algorithms classified the patients according to the medication used (the three-class problem) better compared to the frequency of medication used (the two-class problem). One likely reason for this finding could be the nature of the predicted class in the three-class problem compared to the two-class problem. In the three-class problem, the predicted class was the medicine name, which is a discrete attribute. However, in the two-class problem, the predicted class was the frequency of the medicine use, which is a discrete attribute derived from a continuous attribute (frequency/refill count). On account of the differences between the nature of the predicted classes, it seems that the classification boundary could divide one class from the other in the three-class problem compared to the two-class problem. However, this line of reasoning needs to be further explored for other predicted classes (discrete or continuous) as well as for other datasets.

Overall, we found that the decision tree algorithm performed better compared to a large class of algorithms across both the 2- and 3- class classification problems. Decision tree implicitly performs feature selection using measures like information gain, and they do not require making any assumption regarding linearity in the data [35]. Thus, it seems that decision trees can classify datasets where there are a large number of relevant and irrelevant attributes [28]. Also, due to the superior performance of the decision tree algorithm, we believe that this algorithm could be used for performing machine-learning in healthcare datasets.

Our results have some important real-world implications for using data analytics in the healthcare domain. First, we believe that supervised learning algorithms like decision trees and others can classify medicine intake and the medicine frequency to a certain accuracy. However, as the d-prime values were all less than 1.0 in our results, we believe that these algorithms need more improvements before we can reliably use them for confirming hypotheses in healthcare datasets. Based upon our results, among different algorithms, we would suggest decision trees to be most robust in datasets with a large number of attributes. Second, we also believe that predicting the type of medications consumed and their frequency of use could be extremely helpful for pharmaceutical companies to decide upon their drug manufacture strategies. Specifically, such strategies could reduce supply-chain costs by managing the delays in ordering and stocking of medications.

In this paper, we performed a preliminary analysis on using frequent-set mining approach on a healthcare dataset involving several attributes, and there are a number of extensions possible of this work in the near future. For example, in future, we would like to compare the classification accuracies of ML algorithms after applying PCA and ANOVA and other features selection techniques to those resulting from applying frequent-set mining algorithms. Here, it would be interesting to extend our investigation to other medical and non-medical datasets as well as to other ML algorithms (e.g., neural networks). We plan to embark on some of these ideas as part of our research in the near future.

Acknowledgement. The project was supported by grants (awards: #IITM/CONS/PPLP/VD/03 and # IITM/CONS/RxDSI/VD/16) to Varun Dutt.

References

1. Seeja, K.R., Zareapoor, M.: FraudMiner: a novel credit card fraud detection model based on frequent itemset mining. Sci. World J. (2014)
2. Oswal, S., Shah, G., Student, P.G.: A study on data mining techniques on healthcare issues and its uses and application on health sector. Int. J. Eng. Sci. **7**, 13536 (2017)
3. Parikh, R.B., Obermeyer, Z., Bates, D.W.: Making Predictive Analytics a Routine Part of patient Care. https://hbr.org/2016/04/making-predictive-analytics-a-routine-part-of-patient-care

4. Winters-Miner, L.A.: Seven Ways Predictive Analytics Can Improve Healthcare. Elsevier, New York (2014)
5. Kornegay, C., Segal, J.B.: Selection of Data Sources. Developing a Protocol for Observational Comparative Effectiveness Research: A User's Guide, pp. 109–28. Agency for Healthcare Research and Quality (US), Rockville, MD (2013)
6. Song, F., Guo, Z., Mei, D.: Feature selection using principal component analysis. In: International Conference on IEEE System Science, Engineering Design and Manufacturing Informatization (ICSEM), vol. 1, pp. 27–30 (2010)
7. Surendiran, B., Vadivel, A.: Feature selection using stepwise ANOVA discriminant analysis for mammogram mass classification. Int. J. Recent Trends Eng. Technol. **3**(2), 55–57 (2010)
8. Shlens, J.: A tutorial on principal component analysis. arXiv preprint arXiv:1404.1100 (2014)
9. Kim, H.Y.: Analysis of variance (ANOVA) comparing means of more than two groups. Restor. Dent. Endod. **39**(1), 74–77 (2014)
10. Kumar, M., Rath, N.K., Swain, A., Rath, S.K.: Feature selection and classification of microarray data using MapReduce based ANOVA and K-Nearest Neighbor. Procedia Comput. Sci. **54**, 301–310 (2015)
11. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, vol. 1215, pp. 487–499 (1994)
12. Raghupathi, W., Raghupathi, V.: Big data analytics in healthcare: promise and potential. Health Inf. Sci. Syst. **2**(1), 3 (2014)
13. Sharma, R., Singh, S.N., Khatri, S.: Medical data mining using different classification and clustering techniques: a critical survey. In: IEEE Second International Conference on Computational Intelligence & Communication Technology (CICT), pp. 687–691 (2016)
14. Yadav, C., Wang, S., Kumar, M.: An approach to improve apriori algorithm based on association rule mining. In: IEEE Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), pp. 1–9 (2013)
15. Ilayaraja, M., Meyyappan, T.: Efficient data mining method to predict the risk of heart diseases through frequent itemsets. Procedia Comput. Sci. **70**, 586–592 (2015)
16. Rani, G.U., Prakash, R.V., Govardhan, A.: Mining multilevel association rule using pincer search algorithm. Comput. Sci. **2**(5) (2013)
17. Narvekar, M., Syed, S.F.: An optimized algorithm for association rule mining using FP tree. Int. Conf. Adv. Comput. Technol. Appl. **45**, 101–110 (2015)
18. Tsumoto, S.: Mining diagnostic taxonomy and diagnostic rules for multi-stage medical diagnosis from hospital clinical data. In: IEEE International Conference on Granular Computing. GRC 2007, p. 611 (2007)
19. Kaushik, S., Choudhury, A., Mallik, K., Moid, A., Dutt, V.: Applying data mining to healthcare: a study of social network of physicians and patient journeys. Machine Learning and Data Mining in Pattern Recognition. LNCS (LNAI), vol. 9729, pp. 599–613. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41920-6_47
20. Vembandasamy, K., Sasipriya, R., Deepa, E.: Heart diseases detection using Naive Bayes Algorithm. IJISET-Int. J. Innov. Sci. Eng. Technol. **2**, 441–444 (2015)
21. Gulia, A., Vohra, R., Rani, P.: Liver patient classification using intelligent techniques. (IJCSIT) Int. J. Comput. Sci. Inf. Technol. **5**, 5110–5115 (2014)
22. Parveen, A.N., Inbarani, H.H., Kumar, E.S.: Performance analysis of unsupervised feature selection methods. In: Computing, Communication and Applications (ICCCA), pp. 1–7. IEEE (2012)
23. Danielson, E.: Health research data for the real world: the MarketScan® Databases. Truven Health Analytics, Ann Arbor (2014)

24. KDB+ 3.4: Computer software. Kx Systems, Palo Alto (2016)
25. World Health Organization: Manual of the International Classification of Diseases, Injuries, and Causes of Death, Ninth Revision, Geneva (1977). https://simba.isr.umich.edu/restricted/docs/Mortality/icd_09_codes.pdf
26. Sayad, S.: ZeroR Classifier. http://chem-eng.utoronto.ca/~datamining/dmc/zeror.htm
27. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
28. Mitchell, T.: Decision tree learning. Mach. Learn. **414**, 52–78 (1997)
29. Witten, I., Frank, E., Hall, M.: Data Mining, pp. 102–103. Morgan Kaufmann, Burlington (2010). ISBN 978-0-12-374856-0
30. Langley, P., Sage, S.: Induction of selective Bayesian classifiers. In: Proceedings of the Tenth international Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers Inc., pp. 399–406 (1994)
31. Peng, C.Y.J., Lee, K.L., Ingersoll, G.M.: An introduction to logistic regression analysis and reporting. J. Educ. Res. **96**(1), 3–14 (2002)
32. Brownlee, J.: Logistic Regression for Machine Learning. https://machinelearningmastery.com/logistic-regression-for-machine-learning
33. Hearst, M.A., Dumais, S.T., Osuna, E., Platt, J., Scholkopf, B.: Support vector machines. IEEE Intell. Syst. Appl. **13**(4), 18–28 (1998)
34. Ting, K.M.: Precision and recall. In: Liu, L., Özsu, M. (eds.) Encyclopedia of Machine Learning, p. 781. Springer, New York (2011). https://doi.org/10.1007/978-1-4899-7993-3_5050-2
35. Dezyre: Top 10 Machine Learning Algorithms. https://www.dezyre.com/article/top-10-machine-learning-algorithms/202
36. Piatetsky-Shapiro, G.: Discovery, analysis and presentation of strong rules. In: Knowledge Discovery in Databases (1991)
37. Janecek, A., Gansterer, W., Demel, M., Ecker, G.: On the relationship between feature selection and classification accuracy. In: New Challenges for Feature Selection in Data Mining and Knowledge Discovery, pp. 90–105 (2008)
38. Motoda, H., Liu, H.: Feature selection, extraction and construction. In: Communication of IICM (Institute of Information and Computing Machinery, Taiwan), vol. 5, pp. 67–72 (2002)
39. Pearl, J.: Entropy, information and rational decisions. Technical report. Cognitive Systems Laboratory, University of California, Los Angeles (1978)
40. Russell, S., Norvig, P.: Artificial Intelligence. A modern approach, vol. 25, p. 27. Prentice-Hall, Egnlewood Cliffs (1995)
41. Bayes, M., Price, M.: An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFRS. Philos. Trans. (1683–1775) **53**, 370–418 (1963)
42. Smola, A.J., Schölkopf, B.: A tutorial on support vector regression. Stat. Comput. **14**(3), 199–222 (2004)
43. Wickens, T.D.: Elementary Signal Detection Theory. Oxford University Press, Oxford (2002)
44. Jiang, F., Jiang, Y., Zhi, H., Dong, Y., Li, H., Ma, S., Wang, Y., Dong, Q., Shen, H., Wang, Y.: Artificial intelligence in healthcare: past, present and future. Stroke Vasc. Neurol. SVN **2**, 230–243 (2017)
45. Rajeswari, K., Vaithiyanathan, V., Pede, S.V.: Feature selection for classification in medical data mining. Int. J. Emerg. Trends Technol. Comput. Sci. (IJETTCS) **2**(2), 492–497 (2013)

A Comparative Study to the Bank Market Prediction

Soumadip Ghosh[1], Arnab Hazra[2], Bikramjit Choudhury[3](✉), Payel Biswas[4], and Amitava Nag[3]

[1] Department of IT, Academy of Technology, Aedconagar, Hooghly 712121, West Bengal, India
soumadip.ghosh@gmail.com
[2] Department of CSE, Academy of Technology, Aedconagar, Hooghly 712121, West Bengal, India
arnabhazra08@gmail.com
[3] Department of IT, Central Institute of Technology, Kokrajhar, BTAD 783370, Assam, India
b.choudhury@cit.ac.in, amitavanag.09@gmail.com
[4] Department of Computer Science, Jogesh Chandra Chaudhuri College, Kolkata 700033, West Bengal, India
payel.biswas.cs@gmail.com

Abstract. Bank market prediction is an important area of data mining research. In the present scenario, we are given with huge amounts of data from different banking organizations, but we are yet to achieve meaningful information from them. Data mining procedures will help us extracting interesting knowledge from this dataset to help in bank marketing campaigns. This work introduces analysis and applications of the most important techniques in data mining. In our work, we use Multilayer Perception Neural Network (MLPNN), Decision Tree (DT) and Support Vector Machine (SVM). The objective is to examine the performance of MLPNN, DT and SVM techniques on a real-world data of bank deposit subscription. The experimental results demonstrate, with higher accuracies, the success of these models in predicting the best campaign contact with the clients for subscribing deposit. The performance is evaluated by some well-known statistical measures such as accuracy, Root-mean-square error, Kappa statistic, TP-Rate, FP-Rate, Precision, Recall, F-Measure and ROC Area values.

Keywords: Data mining · Classification
Multilayer Perception Neural Network · Decision tree · Support Vector Machine

1 Introduction

Banks keep huge amount of data about their customers. This data can be used to create and keep clear relationship and connection with the customers in order to target them individually for definite products or banking offers. Usually, the selected customers are

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 259–268, 2018.
https://doi.org/10.1007/978-3-319-96136-1_21

contacted directly through personal contacts, telephone cellular, mail, and email or any other contacts to advertise the new product/service or give an offer, this kind of marketing is called direct marketing. In fact, direct marketing is in the main a strategy of many of the banks and insurance companies for interacting with their customers. Data mining [1] has gained popularity for illustrative and predictive applications in banking processes. Three techniques will apply to the data set on the bank direct marketing. The Multilayer perception neural network (MLPNN) is one of these techniques, which have their roots in the artificial intelligence. MLPNN is a mutually dependent group of artificial neurons that applying a mathematical or computational model for information processing using a connected approach to computation.

Another technique of data mining is the decision tree approach. Decision tree provides powerful techniques for classification [2] and prediction. There are many algorithms to build a decision tree model [3, 4]. It can generate understandable rules, and to handle both continuous and categorical variables. One of the famous techniques of the decision tree is CART, which will be applied in this work. The third technique is Support Vector Machine (SVM) [5], an algorithm for solving the quadratic programming (QP) problem that arises during the training of support vector machines.

2 Related Works

Osuna et al. [6] proved a theorem which suggests a whole new set of QP algorithms for SVMs. By the virtue of this theorem a large QP problem can be broken down into a series of smaller QP sub-problems to converge to the global optimum. This decomposition algorithm can be used to train SVM on larger dataset.

Moro et al. [7] worked with a large dataset, collected over 2008 to 2013 from a Portuguese retail bank, was addressed which includes the recent financial crisis. They analyzed a large set of 150 features related with bank client, product and social-economic attributes. Because of a semi-automatic feature selection explored in the modelling phase of their method, performed with the data prior to July 2012, the data set was reduced to 22 features (which we are using in our approach). They also compared four DM Models (logistic regression, decision trees (DT), neural network (NN) and support vector machine) using two metrics (area of the receiver operating characteristic curve (AUC) and area of the LIFT cumulative curve (ALIFT)) out of which NN presented the best results (AUC = 0.8 and ALIFT = 0.7), allowing to reach 79% of the subscribers by selecting the half better classified clients.

Hu [8] applied data mining techniques to help retailing banks for attrition analysis to identify a set of customer having high probability to attrite. He has used decision tree (DT), boosted naive Bayesian network, selective Bayesian network, neural network as data mining model.

Ling and Li [9] used data mining techniques for direct marketing in three datasets form three different sources. The first dataset was for a loan product promotion in Canada. Second dataset was from a major life insurance company and third dataset was from a company which runs a particular “bonus program”. Two learning algorithms (ADA-boosted Naive Bayes and ADA-boosted C4.5 with CF) that also produce probability had been used.

According to Turban et al. [10] business intelligence includes architectures, tools, databases, applications and methodologies with the goal of using data to support decisions of business managers. Data mining is a business intelligence technology that uses data-driven models to extract useful knowledge i.e., patterns from complex and large dataset [11].

Chitra and Subashini [12] employed some data mining algorithms for customer retention, automatic credit approval, fraud detection, marketing and risk management in banking sector. They have identified some procedures and models to improve customer retention and to fraud detection.

Rafiqul Islam and Ahsan Habib [13] applied DT approach to predict prospective business sector for lending in retail banking. They have used data from different branches of a bank and analysed borrowers’ transactional behavioural data.

3 Dataset Description

The present work is related with direct marketing campaigns of a Portuguese banking institution. We have taken this dataset from the University of California at Irvine (UCI) Machine Learning Repository (as given below in Table 1). The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be (yes) or not (no) subscribed. There are four datasets:

1. “bank-additional-full.csv” with all examples (41188) and 20 inputs, ordered by date (from May 2008 to Nov 2010), very close to the data analyzed.
2. “bank-additional.csv” with 10% of the examples (4119), randomly selected from dataset 1 as mentioned above), and 20 inputs.
3. “bank-full.csv” with all examples and 17 inputs, ordered by date (older version of this Dataset with fewer inputs).
4. “bank.csv” with 10% of the examples and 17 inputs, randomly selected from 3 (older version of this dataset with less inputs).

The smallest datasets are provided to test more computationally demanding machine learning algorithm (e.g. SVM). The classification goal is to predict if the client will subscribe (yes/no) a term deposit (variable y). This UCI dataset is used to evaluate the performances of the multilayer perception neural network (MPLNN), decision tree and SVM classification model. The description of each of the attributes in the dataset is given in Table 1.

Table 1. Attribute description

Sl. no.	Attribute	Meaning	Type
1	age	age of the person	Numeric
2	job	type of job	Categorical
3	marital	marital status	Categorical
4	education	highest education achieved	Categorical
5	default	has credit in default? (categorical: 'yes', 'no')	Binary (Categorical)
6	housing	has housing loan? (categorical: 'yes', 'no', 'unknown')	Binary (Categorical)
7	loan	has personal loan? (categorical: 'yes', 'no', 'unknown')	Binary (Categorical)
8	contact	contact communication type (categorical: 'cellular', 'telephone')	Binary (Categorical)
9	month	last contact month of year	Categorical
10	day_of_week		Numeric
11	duration	last contact duration, in seconds	Numeric
12	Campaign	number of contacts performed during this campaign and for this client	Numeric
13	pdays	number of days that passed by after the client was last contacted from a previous campaign	Numeric
14	previous	number of contacts performed before this campaign and for this client	Numeric
15	poutcome	outcome of the previous marketing campaign	Categorical
16	emp.var.rate	employment variation rate- quarterly indicator	Numeric
17	cons.price. idx	consumer price index- monthly indicator	Numeric
18	cons.conf.idx	consumer confidence index- monthly indicator	Numeric
19	euribor3 m	euribor 3 month rate - daily indicator	Numeric
20	nr.employed	number of employees - quarterly indicator	Numeric
21	Output	has the client subscribed a term deposit? (binary: 'yes', 'no')	Binary (Categorical)

4 Proposed Method

Three techniques namely MLPNN, DT and SVM are applied to the data set for bank marketing prediction. The detailed procedure is divided into two major steps. First one is data pre-processing and then data classification. Figure 1 below depicts the proposed methodology of our system.

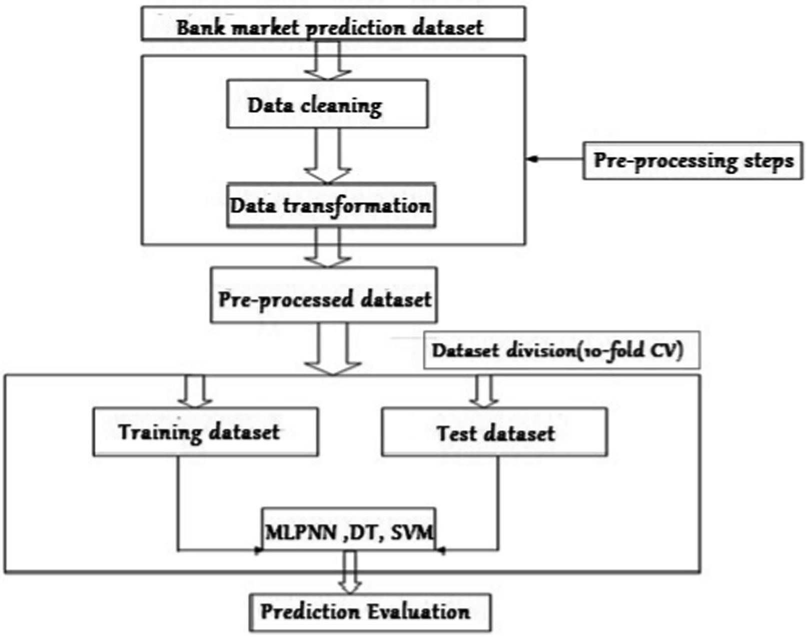


Fig. 1. Proposed methodology of the system using different classifiers

Step 1: Data pre-processing

Data pre-processing techniques are applied to the original dataset before the data classification procedure. It may involve different techniques such as data cleaning and data transformation.

(1a) Data cleaning: Data cleaning denotes the pre-processing of data for removing or reducing noise and the handling the missing values. A missing value is typically replaced by the mean value for that attribute based on statistics. This step is not required in our work as there are no missing or inconsistent values present.

(1b) Data transformation: In data transformation step, the dataset is normalized as because the ANN based technique requires distance measurements in the training phase. It transforms attribute values to a small-scale range like −1.0 to +1.0.

Step 2: Data classification

After pre-processing steps are over, the original data set is divided into two sub-sets namely the training data set and the test data set. We apply 10-fold cross-validation technique for data distribution so as to generate training and test datasets separately. In classification step, firstly the mathematical model of the classifiers is initialized with default control parameters. After initialization is over, they are trained using the training tuples of training dataset. And after the training phase, they are tested with unknown tuples of test dataset as test input to obtain predicted class label. This label is compared with the actual class label to estimate the accuracy of the classifiers being used. The configuration parameters of MLPNN, DT and SVM are given below.

For MLPNN classifier we have,

$$H = \sqrt{I \, . \, O} \tag{1}$$

Where H, I and O denotes the number neurons in the hidden layer, number of input and output attributes respectively. Table 2 describes different metrics of MLPNN model.

Table 2. Different metrics of the MLPNN model

Metric	Value
Number of hidden layers	One
Number of neurons in input layer	Number of input attributes
Number of neurons in output layer	Data classes present
Learning rule	Gradient descent with momentum
Transfer function used	Tan-sigmoid

After construction of the tree, minimal cost complexity pruning algorithm to be used is a post-pruning approach. This algorithm produces a decision tree classifier with minimum cost complexity. Table 3 describes different metrics used in the CART model. All these metrics have their usual meanings.

Table 3. Different metrics of the CART model

Metric	Value
Attribute selection measure	Gini index
Minimal number of instances at terminal nodes	2
Pruning approach used	Post-pruning approach
Pruning algorithm name	Minimal cost complexity pruning
Number of folds used	5
random seed number	1

Different possible combinations like the number of folds used, value of random seed, and different kernel based techniques are investigated here for developing an SVM classifier. Then an SVM model with a Gaussian radial-basis function (RBF) kernel is selected. A non-linear version of SVM can be represented by using a kernel function K as:

$$K\left(x_i, x_j\right) = \phi(x_i) \, . \, \phi\left(x_j\right) \tag{2}$$

Here $\varphi(x)$ denotes non-linear mapping function employed to map the training instances. An SVM model with a Gaussian RBF kernel is defined as:

$$K\left(x_i \cdot x_j\right) = e^{\frac{-\left\|x_i - x_j\right\|^2}{2\sigma^2}} \tag{3}$$

Table 4 below provides different metrics used in the given SVM model. All these metrics have their usual meanings.

Table 4. Different metrics of the SVM model

Metric	Value
Type of kernel used	Non-linear
Kernel name	Gaussian radial-basis function (RBF)
Cache size	250007
Value of σ	0.01
Complexity parameter	1.0
Number of folds used	−1
Random seed value	1
Epsilon value for round-off error	1.0e−12
Tolerance Parameter	0.001

5 Results and Discussion

Here three classifiers namely Multilayer Perceptron Neural Network, Decision Tree and Support Vector Machine are applied to the UCI machine learning repository data set for investigation and performance analysis. Here we have divided the data set into training purpose and testing purpose. The results described here are exclusively based on the simulation experiment that we have taken. We have done several comparison of these classifiers based on some performance measures like classification accuracy, root-mean square error (RMSE) [14], kappa statistic [15] values. And also performed detail accuracy by each class for three classifiers using True Positive Rate (TP-Rate) or Recall, False Positive Rate (FP-Rate), Precision, F-Measure and ROC area values derived from the confusion matrix [16] of each classifier. Three classifiers (MLPNN, DT and SVM) are applied to a test set for classification after completion of the training phase. Firstly, we perform comparisons of these classifiers which are based on said performance measure as shown below in Table 5.

Table 5. Performance comparison of three classifiers

Classifier	Accuracy	Kappa Statistics	RMSE
MLPNN	87.2%	0.7842	0.3018
DT	92.8%	0.8269	0.2678
SVM	86.3%	0.7736	0.3127

From Table 5 we could see that, accuracy of MLPNN, DT and SVM are 87.2%, 92.8% and 86.3% respectively. So it is clear that accuracy wise DT has performed better than MLP and SVM here. Based on the result, DT comes out first with an RMSE value of 0.2678 and a kappa statistic value of 0.8269; followed by MLPNN having an RMSE value of 0.3018 and a kappa statistic value of 0.7842 and SVM stands last with the highest RMSE value 0.3127 and the lowest kappa statistic value 0.7736. Therefore, with regard to the performance measures such as classification accuracy, RMSE and kappa statistic, the DT classifier has performed the best. Figure 2 shows a performance comparison among three classifiers.

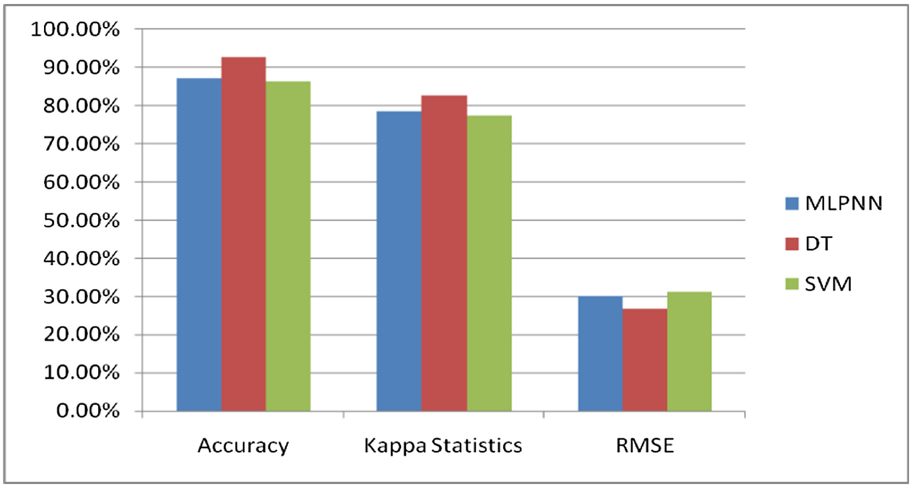


Fig. 2. Performance comparison of three classifiers

After that we have compared these models based on the TP-Rate (or Recall), FP-Rate, Precision, F-Measure and ROC area values derived from the confusion matrix of individual with respect to the test data set.

Table 6. Detailed accuracy by each of the three classifiers

Classifier	TP-Rate/Recall	FP Rate	Precision	F-Measure	ROC Area
MLPNN	87.2%	12.8%	87.2%	87.2%	0.867
DT	92.8%	9.2%	92.8%	92.8%	0.931
SVM	86.3%	13.9%	86.3%	86.3%	0.845

From Table 6 we could discover that the weighted average values of TP-Rate (or Recall), FP-Rate, Precision, F-Measure, and ROC Area for MLPNN classifier are 87.2%, 12.8%, 87.2%, 87.2% and 0.867 respectively; whereas for DT classifier the values are 92.8%, 9.2%, 92.8%, 92.8% and 0.931 respectively. For SVM these values are 86.3%, 13.9%%, 86.3%, 86.3%, and 0.845 respectively. Detail accuracy by the three classifiers is also shown in 2-D column chart in Fig. 3.

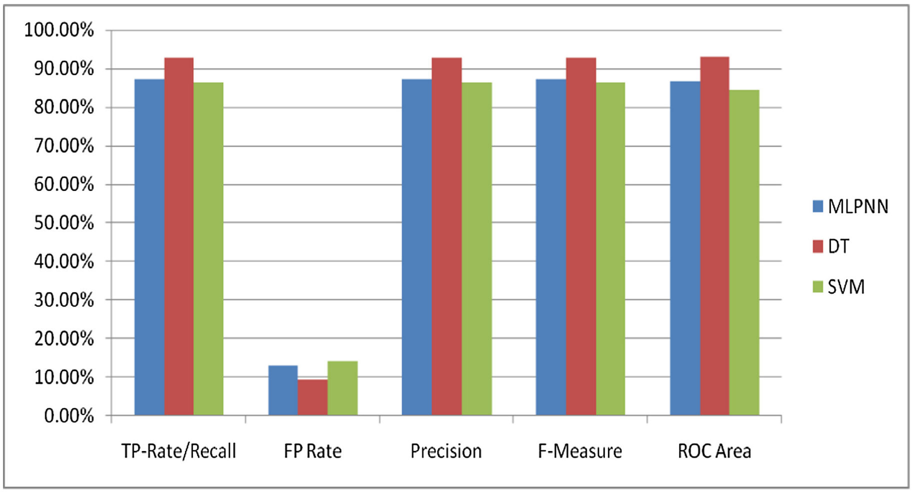


Fig. 3. Detailed accuracy by each of the three classifiers

The DT model has the highest weighted average values for TP-Rate, Precision and F-Measure and the lowest weighted average value for FP-Rate. Indeed, the classification accuracy value of DT model is considerably better (more than 5%) compared to the other models.

6 Conclusion

Bank market prediction is needed for bank deposit subscription, customer relationship management, fraud detection and building marketing strategies. This kind of prediction is certainly helpful for running the business successfully. This paper uses procedures to evaluate and compare the classification performance of three different data mining techniques using models such as MPLNN, Decision tree and SVM on the bank direct marketing dataset. The purpose is to increase the effectiveness of the bank marketing campaigns by identifying the main characteristics that affect the success. These classifiers mainly aim to indicate the deposit money subscribed by the clients. The classification performances of these three models have been evaluated using several useful statistical measures. Finally, experimental results have shown the effectiveness of DT model compared to MLPNN and SVM models. In fact, the accuracy value of DT model is significantly higher (more than 5%) compared to the other classification models used here. So, the DT model developed using CART technique could be very much helpful for direct bank market prediction.

References

1. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, Burlington (2000)
2. Pujari, A.K.: Data Mining Techniques, 1st edn. Universities Press (India) Private Limited, Hyderabad (2001)
3. Quinlan, J.R.: Simplifying decision trees. Int. J. Man Mach. Stud. **27**(3), 221–234 (1987)
4. Breiman, L., Freidman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees Belmont. Wadsworth, Belmont (1984)
5. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
6. Osuna, E., Freund, R., Girosi, F.: An improved training algorithm for support vector machines. In: IEEE NNSP 1997, Amelia Island, FL, pp. 24–26, September 1997
7. Moro, S., Laureano, R., Cortez, P.: Using data mining for bank direct marketing: an application of the CRISP-DM methodology. In: Novais, P., et al. (Eds.), Proceedings of the European Simulation and Modelling Conference – ESM 2011, Guimarães, Portugal, pp. 117–121, October 2011
8. Hu, X.: A data mining approach for retailing bank customer attrition analysis. Appl. Intell. **22**(1), 47–60 (2005)
9. Ling, C.X., Li, C.: Data mining for direct marketing: problems and solutions. In: Proceedings of the 4th KDD Conference, pp. 73–79. AAAI Press (1998)
10. Turban, E., Sharda, R., Delen, D.: Decision Support and Business Intelligence Systems, 9th edn. Prentice Hall Press, Upper Saddle River (2010)
11. Witten, I., Frank, E.: Data Mining – Practical Machine Learning Tools and Techniques, 2nd edn. Elsevier, New York (2005)
12. Chitra, K., Subashini, B.: Data mining techniques and its applications in banking sector. Int. J. Emerg. Technol. Adv. Eng. **3**(8), 219–226 (2013). ISSN 2250-2459, ISO 9001:2008 Certified Journal
13. Rafiqul Islam, M., Ahsan Habib, M.: A data mining approach to predict prospective business sectors for lending in retail banking using decision tree. Int. J. Data Min. Knowl. Manag. Process (IJDKP) **5**(2), 13–22 (2015)
14. Armstrong, J.S., Collopy, F.: Error measures for generalizing about forecasting methods: empirical comparisons. Int. J. Forecast. **8**, 69–80 (1992)
15. Carletta, J.: Assessing agreement on classification tasks: the Kappa statistic. Comput. Linguist. **22**(2), 249–254 (1996). MIT Press, Cambridge
16. Stehman, S.V.: Selecting and interpreting measures of thematic classification accuracy. Rem. Sens. Environ. **62**(1), 77–89 (1997)

Deep Metric Learning for Sequential Data Using Approximate Information

Stefan Thaler[1](✉), Vlado Menkovski[1], and Milan Petkovic[1,2]

[1] Technical University of Eindhoven, Den Dolech 12, 5600 MB Eindhoven, Netherlands
{s.m.thaler,v.menkovski}@tue.nl

[2] Philips Research Laboratories, High Tech Campus 34, Eindhoven, Netherlands
milan.petkovic@philips.com

Abstract. Learning a distance metric provides solutions to many problems where the data exists in a high dimensional space and hand-crafted distance metrics fail to capture its semantical structure. Methods based on deep neural networks such as Siamese or Triplet networks have been developed for learning such metrics. In this paper we present a metric learning method for sequence data based on a RNN-based triplet network. We posit that this model can be trained efficiently with regards to labels by using Jaccard distance as a proxy distance metric. We empirically demonstrate the performance and efficiency of the approach on three different computer log-line datasets.

Keywords: Efficient metric learning · Triplet network · Deep learning

1 Introduction

Metric learning methods enable learning of a distance metric that allows to project data in an embedded space. These methods offer significant flexibility since the data is not mapped to a particular value such as in supervised classification or regression, but are projected to an embedding space based on their relationship to other data points in the dataset. Different kinds of semantical labels can then be assigned, interpretation of the model decision can be grounded in the training data and techniques such as one shot [7] and zero shot [15] learning can be performed. Furthermore, in many scenarios relative differences between data points can be obtained 'cheaper' than direct labels.

In this paper, we examine metric learning on sequence data. We are presented with sequences of symbols (tokens) that are assigned a class or a label. However, our label space is complex and not well defined. There is a large number of classes, and many of those appear infrequently. Certain classes may be considered 'out-of-vocabulary', since they may not appear in the training data, but can appear during inference. In this setting, utilizing all available data requires to collect supervision on the relative distance [9] or ranking [19] between the data points. To tackle this, we propose to use a proxy distance metric that is weakly correlated

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 269–282, 2018.
https://doi.org/10.1007/978-3-319-96136-1_22

to the target metric, and can be used in the ranking task. Learning in such a way can be combined with a supervision signal that allows for high-quality embedding at a fraction of the needed supervision.

More specifically, our contributions are:

- We present a method for efficient metric learning with sequence data by adapting the triplet network for sequence data using recurrent neural networks (RNN).
- We improve the efficiency of the proposed method with regards to the labels by using a proxy distance metric (Jaccard distance) that allows us to learn a high-quality distance metric with small amount of annotations. We detail the approach in Sect. 2.
- We provide empirical evidence on three log-line datasets in Sects. 3 and 4, where we show that the triplet metric learning approach with proxy distance outperforms a RNN model on the same amount of labels.

2 Deep Metric Learning Using Approximate Information

Our goal is to learn an embedding function that maps symbolic token sequences from feature space to a metric space such that semantically similar sequences are mapped close to each other, and non-similar ones are farther apart. The general idea behind this is called metric learning [20]. Recently, deep learning methods based on Triplet networks [19] have been proposed to learn such embedding functions. Triplet networks learn a metric by optimizing a ranking problem on input triplets. A triplet consists of an anchor sequence, a positive sequence, and a negative example. These three examples related to each other by a similarity relationship, i.e., the positive example should be more similar to the anchor example than the negative example. The triplet network is trained on learning a function that embeds examples into a metric space, in which positive sequences are closer to the anchor example than negatives ones.

To the best of our knowledge, triplet networks so far only have been trained using information from labels. That is, an example is positive to the anchor examples if both have the same label, and negative otherwise. Labels are often not available, or labor-intensive to obtain.

The main idea of this paper is to use a proxy distance metric in combination with a few labeled examples to determine the similarity relationship between triplets of examples. This proxy distance metric gives some indication about the ranking but is not very precise otherwise. If needed, the learned metric space can be improved by adding a few labeled examples. Thus, in other words, we conduct a form of weakly supervised learning [22] to learn a distance metric using triplet networks.

This method offers two advantages over simply using a distance metric on the pairwise training examples. First, it scales with the number of training examples, and second, it allows to learn a higher quality, domain-specific metric. It also offers a generic way to combine existing metrics on the input data with labeled examples.

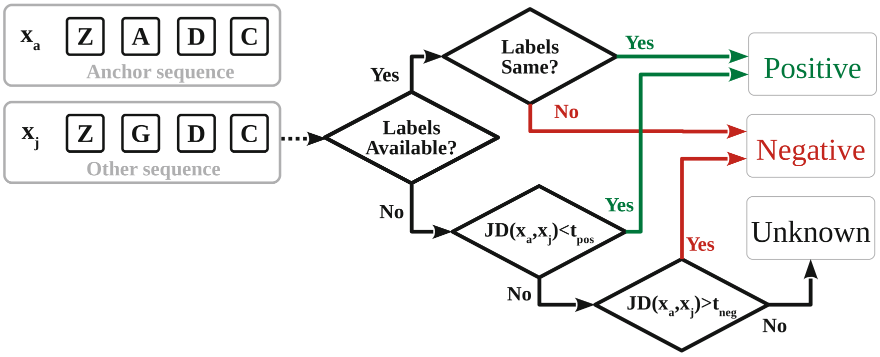


Fig. 1. Here we show the relationship defining processes of a sequence x_j to the anchor sequence x_a. Our main idea is to learn a distance metric in a weakly supervised way by defining the similarity relationships of input data using a proxy distance metric and a few labeled examples. In combination with triplet networks, these similarity relationships allow us to learn a domain-specific metric for our input sequences.

2.1 Triplet LSTM

Commonly, a deep learning algorithm consists of the following four components: a model, an objective or cost function, an optimization procedure, and data. Here we describe the model that we use to learn a distance metric for sequences. We base our model architecture on the Triplet network that was introduced by Wang et al. [19]. Such a model consists of a deep neural network for embedding, followed by a L_2 normalization layer. Each input of the triplet is embedded using the same network, and the normalized output is used for calculating the triplet loss. Instead of a deep convolutional neural network, we use we use an LSTM [5] for embedding the sequences. LSTMs are well-suited for modeling sequences and have been applied to many sequential learning problems, e.g. [8]. We refer to our embedding network including the normalization layer as f, x is a sequence and z is an embedded sequence $z = f(x)$. Figure 2 depicts our triplet network architecture schematically.

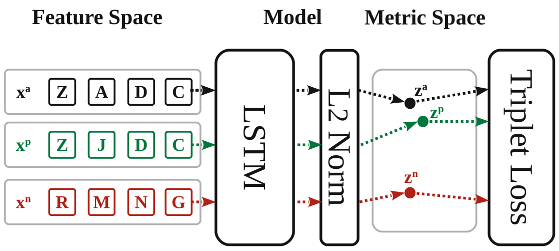


Fig. 2. We use a triplet-LSTM followed by a L_2 normalization layer to learn a metric embedding space for sequences.

2.2 Objective

Here we describe our objective for learning the distance metric for sequences. We use the triplet loss that was introduced by Wang et al. [19]. This objective penalizes triplets where the distance between the embedded anchor and the embedded positive example is larger than the distance between the embedded anchor and the negative example.

Given a triplet of an embedded anchor example z_i^a, an embedded positive example z_i^p and an embedded negative example z_i^n, this objective minimizes the difference in distances between the anchor and the positive and the anchor and the negative example and a given margin α. α is a hyperparameter.

Let T be the set of all possible triplets in a mini-batch, and let $[]_+$ a hinge loss function. Then our learning objective is to minimize the loss L

$$L = \sum_i^N \left[\|z_i^a - z_i^p\|_2^2 - \|z_i^a - z_i^n\|_2^2 + \alpha\right]_+ \tag{1}$$

for the set of all triplets that violate the following constraint:

$$\|z_i^a, z_i^p\|_2^2 + \alpha < \|z_i^a, z_i^n\|_2^2 \; \forall (x_i^a, x_i^p, x_i^n) \in \mathrm{T} \tag{2}$$

The triplet loss L will be small, if the distance between the positive example and the anchor is small and the distance between the negative and the anchor is high and large otherwise. The parameter α ensures that a margin between examples of the same class is allowed.

2.3 Jaccard Distance

We use the Jaccard distance as a proxy metric to determine the similarity relationship between two input sequences. The Jaccard distance JD is a distance measure between two sets x_1 and x_2. It is defined as:

$$JD(x_1, x_2) = 1 - \frac{x_1 \cap x_2}{x_1 \cup x_2} \tag{3}$$

with $\cap$ being the intersection of the two sets and $\cup$ being the union of the two sets. To calculate the Jaccard distance between two token sequences, we treat the sequences as sets of tokens. The Jaccard distance tells us about the diversity of two sequences, but it ignores informative properties of the sequences such as the order of the tokens. We hypothesize that using the Jaccard distance provides sufficient information to rank triplets based on their dissimilarity.

2.4 Method

In this work we propose a method, the relies on a proxy distance metric and a few labeled examples. This method enables to learn a domain-specific distance metric for sequences using a triplet network using only a fraction of the required labels.

To learn distance metrics with a triplet network, we need to define the relationship of input triplet examples. That is, given an anchor example, we need to know whether an example is positive, i.e., belongs to the same class or negative, i.e., belongs to a different class.

The relationship between input examples is defined between an anchor example x_a and another example x_j. We define this relationship using a two-step process. This two-step process is depicted in Fig. 1.

First, if we have information about the labels of both examples, we use this information to determine the similarity relationship. This step is similar to the definition of the relationship in previous works on triplet networks.

If we don't have label information available, we use the Jaccard distance JD as proxy distance metric to determine the relationship between x_a and x_j. If the Jaccard distance is below a threshold t_{pos}, the relationship is *positive* and if it is above another threshold t_{neg}, the relationship is *negative*. If the Jaccard distance is above the positive threshold t_{pos} but below the negative threshold t_{neg}, we define the relationship as *unknown*. If the relationship between a pair of sequences is unknown, we ignore it in the triplet selection process (see Sect. 3.3) of the training phase.

The positive and negative thresholds are hyperparameters and depend on the data domain. The margin between the positive and the negative threshold is meant to increase the accuracy of the approach. That is, only if it is likely that two sequences are similar they should be labeled *positive*, and only if it is very unlikely that they are similar to *negative*.

Formally, we define the similarity relationship between an anchor sequence x_a and another sequence x_j as follows. Let x^a, x^j be two sequences of tokens, l^a, l^j their respective label, $JD(x^a, x^j)$ the Jaccard distance between the two sequences, and t_{pos}, t_{neg} the thresholds for being a positive or a negative example pair. If a sequence is not labeled, it's label is $\emptyset$.

We then define the similarity relationship R between two sequences as:

$$R(x^a, x^j) = \begin{cases} pos, & \text{if } l^a \neq \emptyset \text{ and } l^j \neq \emptyset \text{ and } l^a = l^j \\ pos, & \text{if } l^a = \emptyset \text{ or } l^j = \emptyset \text{ and } JD(x^a, x^j) < t_{pos} \\ neg, & \text{if } l^a \neq \emptyset \text{ and } l^j \neq \emptyset \text{ and } l^a \neq l^j \\ neg, & \text{if } l^a = \emptyset \text{ or } l^j = \emptyset \text{ and } JD(x^a, x^j) \geq t_{neg} \\ unk, & \text{otherwise} \end{cases} \tag{4}$$

with pos being *positive*, neg being *negative* and unk *unknown*.

3 Experimental Evaluation

Here we present our experimental evaluation of our method. We want to show that (i) approximate information in combination with a distance metric is possible, and (ii) that adding labeled examples to this process increases the performance.

We evaluate our approach to the task of classifying log lines from log datasets. Log datasets typically consist of multiple log lines, and each log line can be treated as a sequence of tokens. The task is to classify each log according to the print statement that created a specific log line. Log lines that originated from the same print statement should obtain the same label. Since print statements often have variable parts, the resulting log lines may often differ, and the semi-structured nature of such logs makes this a challenging task.

3.1 Baseline

We use our proxy distance metric as the baseline. For evaluation, we calculate the pairwise Jaccard distance from each log line to each other log line. Given this pairwise distances, we can calculate the accuracy, the true acceptance, and the false acceptance rate (see Sect. 3.9). We refer to our baseline method as *Base-JA*.

3.2 Model

Here we detail the model that we use for performing our deep-learning based experiments. In all our experiments we use the same model. This model takes sequences of token ids as input. These sequence of token ids are mapped to sequences of token vectors using a dense token embedding matrix. The model outputs a point in the metric space in case of the triplet network experiments (see Sect. 3.4) and class probabilities in case of classification experiments (see Sect. 3.5).

We use an LSTM [5,21] to encode the token vector sequences. We use dropout [16] to prevent overfitting, and gradient clipping [12] to prevent exploding gradients. There are many more advanced deep learning architectures for modeling sequences. We chose a very basic architecture because we were mainly interested in studying the effects of combining selecting the triplet with labels. Also, LSTMs have shown to be a reliable choice for modeling sequences [8].

We learn our model parameters using RMSProp [18]. RMSProp is a momentum-based variant of stochastic gradient descent. We train our model in mini-batches. We implemented our model in Tensorflow 1.4.0 [1] and Python 3.5.3.

3.3 Triplet Selection

Selecting the right triplets as input for the triplet network is crucial for the learning to converge [14]. We use an online strategy to select the triples for training our network. We use all valid combinations of anchor-positive and anchor-negative examples per mini-batch to train our network. anchor-unknown example pairs are ignored. Valid in this context means triples that violate the condition in Eq. 2. We additionally add randomly sampled positive examples of each labeled example in a batch to increase the number of valid triplets per mini-batch. These positive examples are uniformly sampled from the distribution of examples of this the same class.

3.4 Experiment - Metric Learning Using Approximate Information

In this experiment, we test our method for learning a metric space using triplet networks and a combination of a proxy distance metric in combination with labeled examples. To embed the sequences in the metric space, we learn an embedding function using the model that was described in Sect. 3.2. The setup for this experiment is schematically depicted in Fig. 2. We refer to this method as *LSTM-Triplet*.

3.5 Experiment - Classification with Augmentation

Triplet networks are complex architectures. To justify the complexity, we compare the results of the metric learning experiments with an experiment that uses the same model (see Sect. 3.2). However, instead of learning a metric space, we learn to classify the sequences based on their label. To do so, we extend our model with an addition softmax layer after the encoding. This layer has neurons according to the number of classes available.

For a fairer comparison, we augment our training data by labeling additional classes using the Jaccard distance. We obtain such further labeled examples for classification by labeling any other unlabeled sequence that has a Jaccard distance to the labeled example that is below a certain threshold with the same class. If there are multiple candidates for labeling, we label the example with the lowest Jaccard distance. We refer to this method as *LSTM-Class*.

3.6 Datasets

To evaluate our method, we use three datasets, a UNIX forensic log file [17], and two system log files of computing clusters BlueGene/L and Spirit2 [11].

The UNIX forensic log file contains 11,023 log lines and was extracted from an Ubuntu system. A forensic log is a log that aggregates many different log sources from a computer system into one large log file. Such an aggregated log file can be used for further forensic analysis in cybercrime investigations. The UNIX forensic log lines originate from 852 print statements, i.e., the logs have 852 different classes. The labels of each log line have been assigned manually by using the source code of the as ground truth. We only use a fraction of the computing cluster system logs. We use 474,700 log lines of the BlueGene/L dataset and 716,577 of the Spirit2 dataset. The BlueGene/L log lines originate from 355 different print statements, and the Spirit2 log lines originate from 691 separate print statements.

Listing 1. Four sample log lines from the UNIX forensic log. Although they are different, they should be labeled the same since they originate from the same print statement.

```
[systemd pid: 1045] : Startup finished in 33ms.
[systemd pid: 867] : Startup finished in 53ms.
[systemd pid: 909] : Startup finished in 34ms.
[systemd pid: 1290] : Startup finished in 31ms.
```

3.7 Data Pre-processing

We pre-process each log line before we use it as input to our deep learning models. We transform the log lines to lower-case and split them into tokens based on whitespace characters. We treat special characters such as brackets also as tokens. All our logs are semi-structured, i.e., they have fixed columns at the beginning and a free text part afterward. In case of the UNIX forensic log, we have removed the fixed columns. In case of the BlueGene/L we have replaced the fixed columns with a fixed tokens and only use the free text part for learning. We replace the information of the fixed columns because it could be easily extracted without a machine learning approach. We add a special start and end token to each sequence. After the pre-processing is done, we have a vocabulary of 4114 different tokens for the UNIX dataset, 101,872 tokens for the BlueGene/L dataset and 59,340 unique tokens for the Spirit2 dataset. Finally, pad the sequences by appending zeros so that each sequence has the same length.

3.8 Hold Out Sets

We use a fraction of 0.2 of the UNIX forensic log and a fraction of 0.1 of the other two datasets as the hold-out set for testing our ideas. The UNIX forensic test set has 602 classes. Each class has two members in the median and the standard deviation of 5.92 member sequences. The BlueGene/L dataset has 245 classes, ten members in the median and a standard deviation of 1162. The Spirit2 dataset has 449 classes with three members in the median and a standard deviation of 1510.

3.9 Evaluation

We compare the validation rate *VAL(d)* and the false acceptance rate*FAR(d)* for the baseline experiment and the metric learning experiment. We use the definition of the validation and false acceptance rate that was introduced by Schroff et al. [14], but we adopt it for sequences. The validation rate measures the correctly classified as same sequence pairs at a given distance threshold, whereas the false acceptance rate measures the incorrectly classified as same sequence pairs at a given distance threshold. We define d as distance threshold, and $D(x_i, x_j)$ as distance between a pair of sequences x_i, x_j. In case of the baseline experiment, $D(x_i, x_j)$ denotes the Jaccard distance between two token sequences, and in case of the metric learning experiment $D(x_i, x_j)$ denotes the squared L_2 distance. We denote all pairs of sequences that are in the same class as P_{same}. We denote all pairs of sequences of different classes as P_{diff}.

True accepts are the sequences that are correctly identified as belonging to the same class at a given distance threshold. We define the set of all *true accepts* $TA(d)$ as:

$$TA(d) = \{(i, j) \in P_{same}, with D(x_i, x_j) \leq d\} \tag{5}$$

False accepts are the sequences that are incorrectly identified as belonging to the same class at a given distance threshold. We define the set of all *false accepts* $FA(d)$ as:

$$FA(d) = \{(i, j) \in P_{diff}, with D(x_i, x_j) \leq d\} \tag{6}$$

The validation rate $VAL(d)$ and the false accept rate $FAR(d)$ for a given distance threshold d are then defined as:

$$VAL(d) = \frac{|TA(d)|}{|P_{same}|}, \; FAR(d) = \frac{|FA(d)|}{|P_{diff}|} \tag{7}$$

Furthermore, we compare the accuracy of the classification experiment. In the Base-JA and LSTM-metric experiments, we report the accuracy at the distance threshold when the validation rate is 1.0, i.e., when all pairs that should be classified as same are classified as same.

3.10 Hyper Parameters and Training Details

We learn our model parameters with a learning rate of 0.01. We decay the learning rate after each epoch with a factor of 0.95. The token embedding matrix has a dimension of 32 and is initialized with a normal distribution and a standard deviation of 0.5. The number of neurons in the LSTM is 32. We clip gradients at 0.5, drop out inputs at a rate of 0.1 and train with a mini-batch size of 100. We use RMSProp with ϵ of 1^{-10} and momentum of 0.0. We train each model for 30 epochs, and randomly shuffle the training data after each epoch. All experiments are conducted with the same hyperparameter settings.

For our LSTM-Triplet experiments, we add two extra positive examples per labeled sequence to each batch. This means that for each labeled sequence we randomly sample two sequences of the same class and add these sequences to the mini-batch. Adding more labeled examples to the batch increases the information available, thus speeds up the conversion of the learning process. Furthermore, we use a Jaccard distance threshold t_{pos} of smaller than 0.3 to determine positive examples, and a Jaccard distance of t_{neg} greater than 0.7 to determine negative examples. We use an α of 0.8 to calculate the hinge loss for our triplet network (see Eq. 1).

We conduct the LSTM-Triplet and the LSTM-Class experiments with 1000, 2500 and 5000 labels. Additionally, we perform the triplet network experiment without additional labels.

4 Results and Discussion

Here we report and discuss the results of our experiments. First, we compare the learned metric spaces. We then report the accuracy of the classification experiments.

In Figs. 3, 4 and 5 depict the VAL-rate against the FAR-rate for increasing distance thresholds. These plots show the results for the baseline Jaccard distance (see Sect. 3.1) and compare it to the triplet network learned with Jaccard distance and 0, 1000, 2500 and 5000 labeled examples (see Sect. 3.4). The results are interpolated and the x-axis of the plots is in log-scale. The bumps in the baseline can be explained by the nature of the datasets. First, the amount of log lines per class varies considerably, and there are few more dominant classes. Second, log lines frequently only differ in very few tokens, i.e., a different variable. The Jaccard distance for sequences is the same for sequences that differ the same amount of tokens, regardless of which class they are.

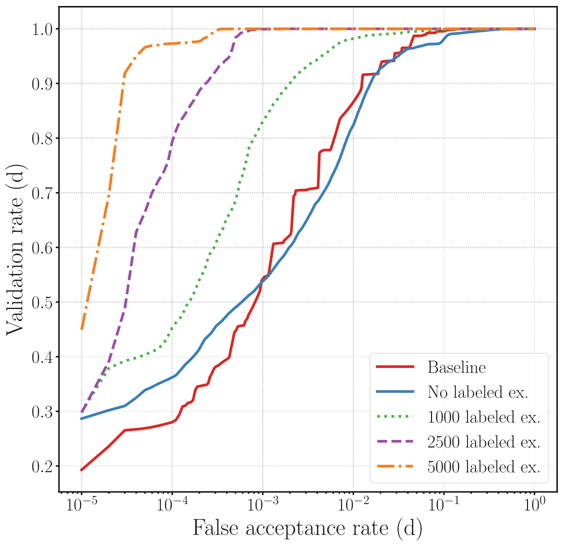


Fig. 3. VAL/FAR rate UNIX forensic dataset. We show the baseline experiments and the triplet network with different amount of additional labeled examples. The bumps in the baseline can be explained by the nature of the datasets.

The performance of the triplet network on the UNIX forensic dataset without additional labels is comparable to the performance of our baseline, the Jaccard distance. In case of the Spirit2 dataset, the performance is better, and in case of the BlueGene/L dataset the performance is slightly worse.

In our triplet network experiments, we use a different positive and negative threshold to determine whether two sequences belong to the same class. That is, only if it is likely that the sequences are equal we mark them as equal, and only

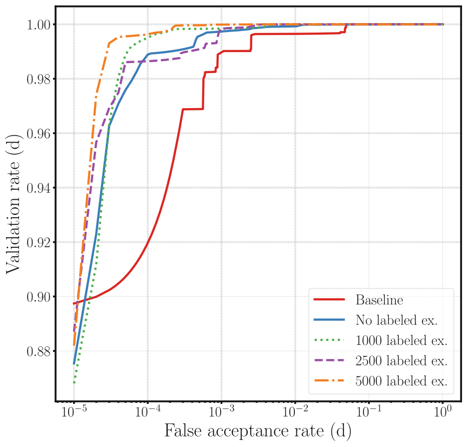


Fig. 4. VAL/FAR rate Spirit2 dataset.

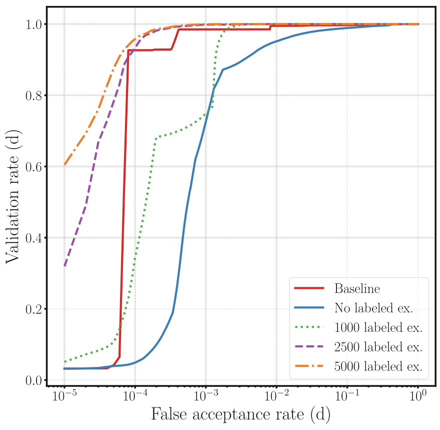


Fig. 5. VAL/FAR rate BlueGene/L dateset.

if it is unlikely that they are equal, we mark them as unequal. One would assume that this leads to a distance metric that is superior to the baseline. However, we can only observe this in case of the Spirit2 dataset.

We argue that this is caused by the way triplet networks learn. We observe that there are consistently many more anchor negative-pairs than anchor-positive pairs during training since more sequences are different to each other than similar. This imbalance leads to a learning behavior that pushes sequences that are different away from each other, but not necessarily closer to each other. This results in well-separated, but split sequence clusters in the metric space.

We hypothesize that it is possible to mitigate this imbalance in negative to positive example pairs by carefully selecting the examples for each batch. However, in a scenario with unknown classes and no labeled data, such a strategy may be difficult to devise.

Furthermore, in Figs. 3, 4 and 5 we can observe that adding labeled examples consistently increases the performance of the metric learning approach. The more labels are added, the better separated the clusters are.

Apart from that, we have also explored the idea of using the triplet network only with fewer labels and discard the information of the Jaccard distance. In this case, the triplet network overfits and performs worse than the baseline and worse than the triplet trained with only the Jaccard distance.

In Table 1 we report the accuracy for our classification experiment (see Sect. 3.5). We calculated the accuracy of the metric-based experiments at the distance threshold d where the VAL-rate was 1.0. We report the distance threshold as well as the accuracy.

In Table 1 we can see that already the baseline Jaccard distance classifies the log lines reasonable well. The LSTM trained with a few labels and augmented with Jaccard distance is consistently better, and the triplet network performs better than the LSTM.

Table 1. We compare the accuracy of our classification experiments. We calculated the accuracy of the metric-based experiments at the distance threshold d where the VAL-rate was 1.0.

Dataset	# labeled examples	Base-JA	LSTM-triplet	LSTM-class
UNIX Forensic	0	**0.841** (0.69)	0.550 (1.40)	N/A
	1000	N/A	**0.866** (1.26)	0.728 (N/A)
	2500	N/A	**0.998** (0.88)	0.913 (N/A)
	5000	N/A	**0.999** (0.68)	0.960 (N/A)
BlueGene/L	0	**0.633** (0.52)	0.612 (0.48)	N/A
	1000	N/A	**0.996** (0.54)	0.971 (N/A)
	2500	N/A	**0.997** (0.52)	0.986 (N/A)
	5000	N/A	**0.999** (0.49)	0.991 (N/A)
Spirit2	0	0.961 (0.51)	**0.990** (0.70)	N/A
	1000	N/A	**0.997** (0.45)	0.973 (N/A)
	2500	N/A	**0.998** (0.45)	0.981 (N/A)
	5000	N/A	**0.999** (0.71)	0.982 (N/A)

We argue that the triplet network performs better in the classification task for two reasons. First, the metric space can deal better with unknown classes. Second, the triplet network can use the available information better. The classification LSTM can only use the information that a sequence belongs to one class and not to all the others. The metric learning LSTM can also utilize the information that is gained when pairwise sequences are from different classes.

5 Related Work

Distance metrics can be learned using Siamese network, and have been originally introduced to distinguish handwritten signatures [2]. They use two identical models and a contrastive energy loss function to learn a distance metric. Siamese networks have been used for deep metric learning in multiple domains, such as face verification [4], image similarity [7] or for learning image descriptors [3].

More recently, Siamese networks have been used to learn distance metrics for sequences and to learn distance metrics for text. Mueller and Thyagarajan propose to use a recurrent Siamese architecture to learn a distance metric for defining similarity between sentences [9]. Neculoiu et al. describe a similar architecture to learn a distance metric between character sequences [10].

Triplet networks have been proposed as an improvement to Siamese architectures [19]. Instead of two identical networks and a contrastive energy function, they learn the distance metric by minimizing a triplet-based ranking loss function. Triplet networks have been successfully applied to a variety of image metric learning tasks [6,14]. Furthermore, magnet networks have been [13] proposed as an improvement over triplet networks. Magnet networks learn a representation space that allows to identify intra-class variation and inter-class similarity autonomously.

6 Conclusion and Future Work

In this paper, we develop a method based on a LSTM triplet network model that allows to efficiently learn a metric space for sequence data. To learn such a metric space, we utilize a proxy distance metric and a limited amount of labels to learn the parameters of an embedding model in a weakly supervised way. Our method provides a general way of combining an existing metric with information obtained from labeled examples. We envision that this method can allow for utilizing domain knowledge to improve the efficiency of metric learning by utilizing domain-specific distance metrics as proxies for various applications.

Furthermore, we hypothesize that the learned metric spaces could be used for interpreting the model's behavior. In this context, we would like to investigate the possibilities for developing interpretations by associating the model's output to labeled training examples.

Acknowledgment. The work presented in this paper is part of a project which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 780495.

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: TensorFlow: a system for large-scale machine learning. In: OSDI 2016, pp. 265–283 (2016)
2. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a "siamese" time delay neural network. In: Advances in Neural Information Processing Systems, pp. 737–744 (1994)
3. Carlevaris-Bianco, N., Eustice, R.M.: Learning visual feature descriptors for dynamic lighting conditions. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014), pp. 2769–2776. IEEE (2014)
4. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 539–546 (2005)
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
6. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: Feragen, A., Pelillo, M., Loog, M. (eds.) SIMBAD 2015. LNCS, vol. 9370, pp. 84–92. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24261-3_7
7. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: ICML Deep Learning Workshop, vol. 2 (2015)
8. Melis, G., Dyer, C., Blunsom, P.: On the state of the art of evaluation in neural language models. arXiv preprint arXiv:1707.05589 (2017)

9. Mueller, J., Thyagarajan, A.: Siamese recurrent architectures for learning sentence similarity. In: AAAI, pp. 2786–2792 (2016)
10. Neculoiu, P., Versteegh, M., Rotaru, M., Amsterdam, T.B.V.: Learning text similarity with Siamese recurrent networks. ACL **2016**, 148 (2016)
11. Oliner, A.J., Stearley, J.: What supercomputers say : a study of five system logs. In: DSN, pp. 575–584. IEEE (2007)
12. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. ICML **3**(28), 1310–1318 (2013)
13. Rippel, O., Paluri, M., Dollar, P., Bourdev, L.: Metric learning with adaptive density discrimination. arXiv preprint arXiv:1511.05939 (2015)
14. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 815–823 (2015)
15. Socher, R., Ganjoo, M., Manning, C.D., Ng, A.: Zero-shot learning through cross-modal transfer. In: Advances in Neural Information Processing Systems, pp. 935–943 (2013)
16. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
17. Thaler, S., Menkovski, V., Petkovic, M.: Unsupervised signature extraction from forensic logs. In: Altun, Y., et al. (eds.) ECML PKDD 2017. LNCS (LNAI), vol. 10536, pp. 305–316. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-71273-4_25
18. Tieleman, T., Hinton, G.: Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA: Neural Netw. Mach. Learn. **4**(2), 26–31 (2012)
19. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1386–1393 (2014)
20. Xing, E.P., Jordan, M.I., Russell, S.J., Ng, A.Y.: Distance metric learning with application to clustering with side-information. In: Advances in Neural Information Processing Systems, pp. 521–528 (2003)
21. Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent neural network regularization. arXiv preprint arXiv:1409.2329 (2014)
22. Zhou, Z.H.: A brief introduction to weakly supervised learning. Natl. Sci. Rev. **5**, 44–53 (2017)

Machine Learning Applied to Point-of-Sale Fraud Detection

Christine Hines and Abdou Youssef

George Washington University, Washington, DC 20052, USA
{chines57, ayoussef}@gwu.edu

Abstract. This paper applies machine learning (ML) techniques including neural networks, support vector machines Random Forest, and Adaboost to detecting insider fraud in restaurant point-of-sales data. With considerable engineering of the features, and by applying under-sampling techniques we show that ML techniques deliver very high fraud-detection performance. In particular, RandomForest can achieve 91% or better across all metrics when using a model trained on one restaurant to detect fraud in a separate restaurant. However, there must be sufficient fraud samples in the model for this to occur. Knowledge and techniques from this research could be used to develop a low-cost product to automate fraud detection for restaurant owners.

Keywords: Machine learning · Classification · Outlier detection
Fraud detection · Point-of-sale data

1 Introduction

Occupational fraud costs the average organization 5% of revenues per year according to survey participants in the Association of Certified Fraud Examiners (ACFE) 2017 Global Fraud Study [1]. Using the 2017 Gross World Product as a basis, this translates to a potential annual loss of ~$3.8 trillion due to insider fraud.

Restaurants are especially sensitive to the 5% impact on sales as they have one of the lowest profit margins of any industry, often in the 3–6% range. And as more than 7 out of 10 restaurants are individually owned [2], most restaurants do not have the resources to invest in costly auditing systems to detect and prevent fraud. Yet according to ACFE, the smallest organizations tend to suffer disproportionately large losses due to occupational fraud [1]. Because of the prevalence of insider fraud and the scant financial resources of most restaurants, a low-cost, automated solution to fraud detection is critical to the livelihood of restaurant owners, yet fraud detection is not available in the majority of point-of-sale (POS) systems. The POS data used in this research was provided to us from four different restaurants for the purpose of determining if machine learning techniques can detect insider fraud.

In this paper, we demonstrate that it is possible to detect even very small amounts of fraud with machine learning using the same fraud detection model across data from multiple restaurants. However, a large amount of data pre-processing time is required to achieve success. As the details of the research show, a large amount of time must be spent building deep knowledge of the data, determining how fraud presents in the data,

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 283–295, 2018.
https://doi.org/10.1007/978-3-319-96136-1_23

and engineering features to enable the algorithms to detect fraud. The results of our work show that it is possible to achieve the high levels of accuracy and precision necessary for the models to be used by independent restaurant owners for detecting fraud.

The paper is organized as follows. Section 2 explains the challenges and difficulties of conducting research on restaurant point-of-sale data. The scope of the work, presented in Sect. 3, explains both how a subset of algorithms were selected and which types of fraud were chosen for study from among numerous fraud techniques used by restaurant employees. Section 4 presents the methods and tools used in this research. The results of the research are presented and discussed in Sect. 5. Finally, Sect. 6 outlines directions for future research extensions.

2 Challenges and Related Work

Research on restaurant fraud is challenging for a number of reasons. First, fraud detection is a special case of outlier detection, and results of the outlier literature review show that while there are techniques to improve the performance of machine learning algorithms such as over- and under-sampling, outlier detection remains a current challenge and an area of study within the machine learning field. This paper leverages a number of research papers on anomaly and outlier detection techniques [11, 14, 16] as well as a couple on sampling techniques [10, 12]. This paper applies both over and under-sampling techniques. While not using the same feature engineering techniques as in [14] this paper demonstrates and reinforces the importance of engineering features to detect anomalies.

Second, there is a dearth of papers on fraud detection in restaurant POS data. One of the newest POS systems offers a fraud detection module [16], but information about methods, techniques, and any details about how this is accomplished are proprietary and unpublished. Some large restaurants and retail clothing stores have applied cameras to detect cashier fraud [9], but detecting fraud in images and video requires a relatively high cost to implement, and cannot cover certain kinds of fraud. Therefore, camera-based fraud detection is not a viable solution for most restaurant owners, and hence will not be pursued in this research effort. However, there are many published papers that demonstrate the ability of machine learning algorithms to detect credit card fraud [3, 7, 8, 13]; this paper extends this research to show these algorithms can also detect fraud in POS data.

As no papers were found on the topic of check fraud detection, all of our research on engineering features to detect check fraud is novel. References [11, 12] explain the importance of feature engineering, but do not provide specific examples of features as will be provided in this paper.

Third, individually owned restaurateurs lack the time and resources to initiate or participate in research, and are highly averse to sharing data with outsiders. Fourth, due to the fact that most restaurants lack auditing systems and processes, and there is a high turnover of restaurant employees, most fraud goes undetected. As a result, examples of known fraudulent data are difficult to obtain.

3 Scope

The scope of this research will be limited by selecting a subset of machine learning techniques and types of fraud.

3.1 Fraud Types

As published in numerous restaurant industry online articles and books such as [6] there are many ways that restaurant employees can commit fraud against the restaurant. As the data provided for this research only includes point-of-sale data and not inventory or accounting information, only some of the fraud types can be detected by the data provided to this research project. The two fraud types selected for study were "Bartender No Sale", and "Server Rotating Check Item" as described in Table 1. However, the development and study of classifiers for "Bartender No Sale", which requires the detection of missing transactions, is still in the process. Therefore, only classification results for "Server Rotating Check Item" will be included in this paper.

Table 1. Fraud methods.

Name	Description
Bartender No Sale	Sell a drink, collect cash from the customer, use "no sale" button to open cash register and make change. Sale is not recorded in the POS system
Server Rotating Check Item	After a customer orders a commonly occurring check item, such as a Coke, and is ready to pay, print the check for the customer. If the customer pays in cash, before closing the check, transfer the check item to another check and pocket the cost of the item

3.2 Fraud Types

The algorithms studied in this research were selected by reviewing the top performing machine learning and data mining algorithms as applied to credit card fraud detection. Phua et al. [5] published a comprehensive survey on data mining-based fraud detection research. Based on speed and accuracy considerations, and to cover a cross-representation of approaches to classification, we selected the algorithms listed in Subsect. 4.4.

4 Data, Methods and Tools

This section presents a description of the data, data pre-processing, labeling, and methods and tools used to apply machine learning to the problem at hand.

4.1 Data Profile

An industry expert who has been implementing restaurant POS systems for over 20 years provided the data and expertise on restaurant fraud. Data was provided for four restaurants spanning an average of four years of data per restaurant. Primary types of information received for each of the restaurants include: ordered items; payments; discounts employees; items; table types; payment types; revenue centers; tax groups; and price levels. Each restaurant had a slightly different configuration for the POS system. Differences in configurations contribute to the complexity of pre-processing and interpretation of the results. Differences in configurations also translate to a higher cost of developing an automated fraud detection system. Examples of differences include different job ids for the same job title, different seating area ids for the bar area, difference customizations of some of folder name, i.e. yyyymmdd versus mmddyyyy.

Tables 2, 3 and 4 show the number of checks, ordered items, and fraud examples by restaurant and per year. While the number of examples labeled as fraud represent less than 1% of the ordered items, it should be recalled that the labeling was not meant to catch all fraud, only those that were highly suspicious. So it is likely that the actual amount of fraud is much higher, possibly as much as 10% higher which would translate to 1% at Restaurant 1, 2% at Restaurant 2 and still less than 1% at Restaurant 3.

Table 2. Restaurant 1 data set size by year.

Record type	Year			
	2011	2012	2013	2014
Checks	32,702	50,966	52,483	19,382
Ordered items	249,378	407,688	434,518	167,054
Fraud	253	769	968	430

Table 3. Restaurant 2 data set size by year.

Record type	Year			
	2012	2013	2014	2015
Checks	25,685	32,262	31,499	7,597
Ordered items	343,056	424,325	420,050	101,878
Fraud	98	162	156	36

Table 4. Restaurant 3 data set size by year.

Record type	Year				
	2009	2010	2011	2012	2013
Checks	10,625	11,273	11,216	11,842	10,272
Ordered items	93,582	99,519	98,510	109,425	97,645
Fraud	87	69	46	103	70

4.2 Data Pre-processing

At its core the dataset was built from the ordered items table. Ordered items that have a price of $0 were filtered out. Examples of ordered items with $0 value are condiments such as ketchup and mustard, toppings such as tomato and onion, tap water, and side orders where the price is included in the price of the entrée, e.g. when French fries or a side salad can be selected as a side dish to a hamburger, and special order notes, such as "over easy", "no mushrooms". All of the data was standardized and scaled before use to ensure that the fields with larger numeric values did not bias the results.

4.3 Fraud Class Labeling

While two types of fraud are being studied, for ease of interpreting and improving the results, it was decided to analyze them separately. For this paper, the data was only labeled for "Server Rotating Check Item" type fraud. This allowed the data to be labeled as fraud or non-fraud for binary classification. After more knowledge is gained about fraud detection in POS systems, it may be possible to refine the labels for a multiple class classifier (with each type of fraud labeled as a separate class) for improved process automation.

As pointed out in [5], finding and labeling fraudulent data is impossible to do with absolute certainty. Due to the inability of any of the restaurant owners to point to specific transactions and to state with certainty that they were fraudulent, a threshold of less than 100% certainty was set for labeling. The threshold can be described as "highly likely". Stated another way, if the transactions labeled as fraud were shown to a restaurant owner, they would cause the owner to question the trustworthiness of that employee, but not immediately assume that those transactions involve fraud for sure. Given the availability of the POS consultant, a supervised process was used to label the data.

4.4 Algorithm Selection

The algorithms selected for this research came from the algorithms that have been used successfully in credit card fraud detection, as there is no published research for detecting fraud in restaurant point-of-sale data. Of those algorithms, we decided to select one from each of the most common families of classifiers, decision trees (RandomForest), probabilistic classifier (naivebayes), artificial neural network (neuralnet), k-nearest neighbor, and linear/kernel-based classifiers (support vector machine), so it can be determined if one type of algorithm performs best for this data domain.

4.5 Principal Component Analysis

For the purpose of feature set reduction, principal component analysis (PCA) is sometimes applied in machine learning if there is a large number of features and high level of correlation between the features. If applicable, a graph of the standard deviations from the principal components can be used to identify the correct cutoff point for the number of principal component vectors that should be used.

The coefficients from the principal component rotations are used as a matrix and multiplied against the dataset to generate a new matrix with artificial features. The number of artificial features will be equal to the number of principal component rotation vectors that are used. For example, if there were 42 features in the dataset and 15 principal component rotation vectors were used, the dataset would be mapped to a new dataset with the same number of records, but with only 15 artificial features instead of 42 features. The artificial features are then used to train the classifiers. Similarly, the test dataset will need to be transformed by the same principal component matrix of vectors to a new vector space with artificial features. Reference [4] explains the application of PCA to machine learning in more detail.

4.6 Tools

A Macbook Pro with a 2.53 GHz Intel Core 2 Duo processor and 8 GB memory was used for all aspects of the research. A laboratory version of Mysql 8.0 is being used despite missing key features because it supports a recursive common table environment (CTE) SQL structure and capability that was necessary for the queries.

Data mining algorithms used in this research are standard R libraries listed below.

- K Nearest Neighbor (KNN) - knn from library “class”
- Naïve Bayes (NB) - naiveBayes from library “e1071”
- Neural Network (NN) - neuralnet from library “neuralnet”
- RandomForest (RF) - ensemble decision tree classifier, library “randomForest”
- Support Vector Machine (SVM) with different kernels - svm from library “e1071”
- Adaboost from library “fastAdaboost”.

4.7 Definitions

Native Features: Features present in the data as it was generated by the POS system.
Engineered Features: Features derived from native features based on an understanding of the data domain to enhance the classification performance. More details about engineered feature are provided in Sect. 5.2.
Artificial Features: Features created by transforming data into a different vector space, using PCA to map the data from native features space to artificial features.
Check: A restaurant check refers to the group of menu items ordered by the customers at a particular table.
Menu Items: A complete list of all menu items available to be ordered by a customer.
Menu Item Types: Each menu item has a menu item type. Examples of menu item types are appetizers, beverages, dessert, entrée, and other.
Ordered Items: Items ordered from the menu by a customer and recorded on the check.

5 Research Results

In summary, the results will show that when training data contained a sufficient number of fraud samples, some of the algorithms were able to successfully detect fraud when tested on a different restaurant.

5.1 Statistical Analysis

Concurrent with the class labeling was the statistical analysis of the data. This was necessary to set the thresholds for labeling of "no sale" fraud class. The only way to determine if checks were missing was to determine the average number of checks per server per season, per day of the week, per shift, and then look for servers that had far fewer checks paid in cash than other servers. Therefore average statistics were generated.

After comparing the actual check counts per shift with the average check counts per shift for checks paid in cash, the variance was so small in most cases that either "bartender no sale" fraud was not being committed at the restaurant being studied or it was too small to detect. There were two shifts in the data where the variance was great enough to be confident that this type of fraud was being committed, but it was determined that that was too small of a class size to be used with machine learning algorithms. And only one restaurant had bar transactions tracked as a separate serving area, which would make it very difficult if not impossible to detect this type of fraud in the data from the other restaurants.

So the research focus diverted from studying "bartender no sale fraud". However, the value of engineered features for fraud detection was conveyed to the selection and development of features for detecting "server rotating check item" fraud. And a hypothesis was informally formed that the detection of fraud may in fact not be possible without the use of engineered features.

5.2 Feature Selection and Engineering

Feature selection for "server rotating check item" was an iterative process that was concurrent to, and driven by, the fraud data labeling process. For every question that had to be answered for the POS consultant, the features that were required to answer the question were added to the feature set that would be required to detect the fraud. As the features required to answer the questions did not exist natively in the dataset, they had to be generated from the data that did exist.

A key aspect of detecting this type of fraud is determining the menu item type of the frequently ordered item that was used to commit the fraud. Frequently ordered items can fall into any of the menu item types except for entrée. Menu Item Types include appetizers, beverages, dessert, entrées, other.

In addition, it is necessary to determine the type and number of menu items on the originating check, and the number and type of ordered items that are included on the check to which the ordered item is transferred. So in addition to the basic calculated features, such as check paid hour, and length of time the check was open, 32 additional engineered features were developed to provide the context for the transferred menu items on the check. A total of six sets of features were used in this research; 46 native features that existed in the original dataset, 46 engineered features which are a combination of 14 native feature and 32 engineered features, artificial features sets of 15 and 32 were generated using the PCA rotations from the native features, and artificial feature sets of 11 and 35 were generated from the engineered features.

5.3 Algorithm Parameter Tuning

Preliminary training and testing runs of the algorithms were conducted on the data to determine the best parameter settings for each of the algorithms included in the research.

SVM was tested with radial, linear, and polynomial kernels over a range of cost and gamma settings. The gamma setting only applies for a radial kernel and is a measure of the width of the kernel as explained in [20]. Larger radial kernels will have smaller gammas. In addition, the polynomial kernel was tested over degrees 1 through 4. The parameters providing the optimal results for SVM were type = "C", kernel = "polynomial", degree = 3, gamma = 0.1, cost = 1 so these are the parameters used to generate the results. However, in case for testing Restaurant 1 2013 as the performance was below par additional tuning runs were conducted and it was found that kernel = "radial" provided better results for only that case. In all other cases, "polynomial" provides better results. NN was tested with one and two hidden vertices within each layer across a range of repetitions from 1 to 20. With two hidden vertices, the algorithm took over 2 h even for the small sample size of 20,500, which was considered unacceptable. Therefore one hidden vertex and 3 repetitions were used because they provided sufficiently high accuracy and precision in most cases within the processing power of the computer used to support the research. Also linear.output = FALSE, err.fct = "ce", likelihood = TRUE was used so NN would perform as a binary classifier. For KNN, k, the number of neighbors used to determine the class assigned to the example, was tested over a range of 1 to 5. The results were within $\pm 1\%$, demonstrating that as expected the vector space of the fraud class examples are relatively well separated from the non-fraud vector space and therefore insensitive to a change in k over the range of 1 to 5. So for performance reasons, k = 1 was used. Default settings were used for RF and NB.

5.4 Algorithm Results

Training and Testing Datasets. The classifiers were trained and tested using randomly sampled data from Restaurants 1, 2, and 3. Restaurant 4, being a 7–11 type restaurant had a much simpler type of fraud being perpetrated that could be detected without the use of machine learning. SVM and Random Forest algorithms were configured specifically for binary classification.

Most machine learning experiments rely on a training set larger than the test set with a ratio of 4/5 for training and 1/5 for testing. This ratio would not work well because of the extreme imbalance of the classes (relatively very few fraud cases). Therefore 2/3 was used for training and 1/3 for testing. In addition, due to multiple hour run times for each algorithm and the accepted practice of applying under-sampling in the case of imbalanced classes, the non-fraud class was randomly sampled to a subset of 30,000 before being split into training and testing datasets. All fraud examples were then split into training and test datasets. The number of records used for training and testing is shown in Table 5.

Table 5. Training and testing dataset runs 1, 2, 3.

Run	Purpose	Source	No fraud	Fraud	Fraud percent
1	Train	Rest. 1 2012	20,000	769	3.8%
	Test	Rest. 1 2013	10,000	968	9.7%
2	Train	Rest. 1 2012	20,000	769	3.8%
	Test	Rest. 2 2013	10,000	162	1.6%
3	Train	Rest. 1 2012	20,000	769	3.8%
	Test	Rest. 3 2009	10,000	87	0.9%

Performance Metrics. The weaknesses of using accuracy, recall, and precision to evaluate machine learning algorithms has been pointed out in many papers as explained in [5]; however, these metrics are still commonly used for the comparison of machine learning algorithms and are used in fraud comparison papers such as [3]. So for ease of comparison, accuracy, precision, and recall, defined in Eqs. 1, 2 and 3 below were used as performance metrics. In the definitions, true positive refers to a fraud correctly detected by the classifier; true negative refers to a non-fraud correctly so reported by the classifier, and false positive refers to a non-fraud that is incorrectly reported as fraud by the classifier.

$$Accuracy = \frac{(\#True\,negatives + \#True\,positives)}{\#All\,examples} \tag{1}$$

$$Recall = \frac{\#True\,positives}{\#All\,true\,positives} \tag{2}$$

$$Precision = \frac{\#True\,positives}{\#True\,Positives + \#False\,positives} \tag{3}$$

Accuracy was included for thoroughness, but as can be seen from the results, because fraud cases account for such a small percentage of the dataset, with one exception (NB with native features), accuracy is always close to 100% regardless of how many fraud examples are actually detected.

The recall and precision metrics have special meaning when applied to fraud detection because they concern the lives and careers of people. Recall translates to the number of times that fraudulent behavior was detected from among all the times that someone actually committed fraud. Precision shows how well the algorithms avoided falsely accusing someone of fraud. So for insider fraud detection, it is usually considered more important to avoid false accusations than to ensure that all fraudulent examples are detected. Therefore, when improving performance of the algorithms, preference should be given to improving precision over recall.

Native Features Performance. As shown in Table 6, all of the algorithms except neural net detected fraud with the native features. However, only RF achieved above 90% in precision and all of them were except NB were below 40% for recall. NB had a relatively high recall at 88%, but the accuracy was only 39.5% – the lowest of any algorithm for any of the feature sets. So none of the algorithms performed well enough with native features and results with artificial features mirrors these results.

Table 6. Native and engineered feature results for training and testing with Restaurant 1.

Algorithm	Native features			Engineered features		
	Accuracy	*Recall*	*Precision*	*Accuracy*	*Recall*	*Precision*
KNN	97%	36%	37%	100%	97%	91%
NB	40%	88%	4%	99%	100%	75%
NN	97%	0%	0%	99%	100%	75%
RF	98%	20%	92%	100%	91%	96%
SVM	97%	35%	51%	100%	99%	95%

Engineered Features Performance. For cross validation as shown in Table 6 KNN, SVM, and RF achieved over 90% across all metrics with NB and NN close behind with all results being higher than 75%. So the identification and generation of engineered features was critical to the success of detecting "server rotating check item" fraud in this restaurant POS dataset.

Results for training and testing on different restaurants are shown in Tables 7, 8, and 9. The results demonstrate that a model can be developed on one restaurant and successfully applied to another restaurant. When training and testing on the same restaurant ADA and RF performed the best. When training and testing on different restaurants the top algorithm varied. In all scenarios the algorithms had difficulty identifying fraud in Restaurant 3.

Table 7. Engineered feature results for machine learning trained on Rest. 1 2012 data.

Alg.	Test with Rest 1 2013			Test with Rest 2 2012			Test with Rest 3 2009		
	Acc.	*Rec.*	*Prec.*	*Acc.*	*Rec.*	*Prec.*	*Acc.*	*Rec.*	*Prec.*
KNN	**99%**	92%	95%	99%	89%	78%	**~100%**	74%	79%
NB	98%	99%	86%	99%	**~100%**	54%	99%	99%	48%
NN	98%	**~100%**	84%	99%	**~100%**	58%	99%	**100%**	52%
RF	**99%**	93%	**98%**	**~100%**	91%	91%	99%	34%	**81%**
SVM	**99%**	86%	**98%**	**~100%**	88%	88%	**~100%**	92%	71%
ADA	**99%**	92%	**98%**	99%	45%	94%	99%	10%	32%

Table 8. Engineered feature results for machine learning trained on Rest. 2 2013 data.

Alg.	Test with Rest 1 2013			Test with Rest 2 2012			Test with Rest 3 2009		
	Acc.	*Rec.*	*Prec.*	*Acc.*	*Rec.*	*Prec.*	*Acc.*	*Rec.*	*Prec.*
KNN	**96%**	**55%**	**96%**	**~100%**	97%	79%	**~100%**	82%	71%
NB	92%	11%	77%	99%	**~100%**	60%	99%	84%	59%
NN	91%	0%	0%	98%	0%	0%	99%	0%	0%
RF	91%	0%	0%	**~100%**	**~100%**	94%	**~100%**	**99%**	**91%**
SVM	91%	0%	0%	98%	6%	91%	99%	7%	86%
ADA	91%	0%	0%	**~100%**	95%	**~100%**	87%	75%	45%

Table 9. Engineered feature results for machine learning trained on Rest. 3 2009 data.

Alg.	Test with Rest 1 2013			Test with Rest 2 2012			Test with Rest 3 2009		
	Acc.	*Rec.*	*Prec.*	*Acc.*	*Rec.*	*Prec.*	*Acc.*	*Rec.*	*Prec.*
KNN	96%	45%	95%	**~100%**	62%	85%	**~100%**	83%	77%
NB	**99%**	**97%**	86%	99%	**99%**	53%	98%	**93%**	27%
NN	93%	0%	0%	99%	0%	0%	99%	0%	0%
RF	93%	0%	0%	**~100%**	63%	**~100%**	Algorithm didn't complete		
SVM	93%	1%	**~100%**	**~100%**	92%	93%	**~100%**	49%	**89%**
ADA	93%	0%	0%	**~100%**	79%	**~100%**	**~100%**	83%	85%

Artificial Features Performance. Runs were done with artificial features derived from either native or engineered features, using the Restaurant 1 dataset for training and testing. Using the standard deviation PCA graph for native features, two knees were identified; the first at 15 and the second at 32. Runs were done with both sizes of artificial features, and as expected the results were equally weak as for native features alone. Using the standard deviation PCA graph for engineered features, also two knees were identified; the first at 11 and the second at 35. With 11 artificial features, all algorithms delivered roughly the same precision and recall as they did with the full 42 engineered features, proving that PCA can be successfully used for feature reduction. Results for 11 and 35 features are shown in Table 10.

Table 10. Artificial feature results from engineering features and training with Restaurant 1 2012 and testing with Restaurant 1 2013.

Alg.	Eng. Features			Art. Features 11			Art. Features 35		
	Acc.	*Rec.*	*Prec.*	*Acc.*	*Rec.*	*Prec.*	*Acc.*	*Rec.*	*Prec.*
KNN	**~100%**	97%	91%	98%	82%	95%	**99%**	88%	95%
NB	99%	**~100%**	75%	**99%**	**~100%**	90%	**99%**	**~100%**	89%
NN	99%	**~100%**	75%	**99%**	**~100%**	90%	98%	**~100%**	84%
RF	**~100%**	91%	**96%**	97%	73%	96%	**99%**	89%	**97%**
SVM	**~100%**	99%	95%	95%	46%	**97%**	97%	70%	96%

6 Conclusion

For training and testing on the same restaurant when there was sufficient fraud samples in both the training and testing datasets as there is for Restaurants 1 and 2 all algorithms delivered 80% or better across all metrics. The best performing algorithm across when training and testing on different restaurants with 71% to 99% in 15 of the 18 metric cases was KNN. SVM, NB, and RF delivered 70% or better for 14, 13, and 12 (respectively) of the 18 cases. RF provided the best performance overall with over 90% for precision, accuracy, and recall in 17 of 27 metric scenarios. However, the fact that RF took 40–60 min to run despite the relatively small dataset may prove problematic for practical use by restaurant owners.

7 Future Research

Opportunities for additional research are many as there is a lack of public research on this topic. As the algorithms did not perform quite as well when the training was done on one restaurant and tested on another, an interesting next step would be to compare the results when the training and testing is done independently for each restaurant. Deep learning has been applied to detect other types of fraud could be applied to this dataset if more data could be obtained or synthetically generated data to create a larger labeled dataset. Other feature reduction techniques could be applied to determine if the execution time of RF could be reduced. And finally there is the opportunity to more fully automate the components of this research so that restaurant owners could use it in practice.

Acknowledgement. Data and expertise on normal restaurant server practices, and restaurant fraud was provided by an industry expert with over 20 years experience selling, installing, upgrading, troubleshooting, and providing training for POS systems in multiple geographic regions within the United States.

References

1. Association of Certified Fraud Examiners (ACFE): Report to the nations on occupational fraud and abuse (2017)
2. National Restaurant Association: 2016 Restaurant operations report (2016)
3. Bhattacharyya, S., Jha, S., Tharakunnel, K., Westland, J.C.: Data mining for credit card fraud: a comparative study. In: 14th International Conference on Security and Cryptography, SECRYPT 2017 (2017)
4. Jotheeswaran, J., Loganathan, R., Madhu Sudhanan, B.: Feature reduction using principal component analysis for opinion mining. Int. J. Comput. Sci. Telecommun. **3**(5), 118–121 (2012). ISSN 2047-3338
5. Phua, C., Lee, V., Smith, K., Gayler, R.: A comprehensive survey of data mining-based fraud detection research. In: 2010 International Conference on Intelligent Computation Technology and Automation, pp. 11–12, May 2010

6. Whitfield, C.A.: Guess who's eating your profits: the manager's essential guide to restaurant and bar loss prevention and investigations. AuthorHouse, 23 April 2013. ISBN-10: 1481725130
7. Patidar, R., Sharma, L.: Credit card fraud detection using neural network. Int. J. Soft Comput. Eng. (IJSCE) **1**, 32–38 (2011). ISSN 2231-2307
8. Sahin, Y., Duman, E.: Detecting credit card fraud by decision trees and support vector machines. In: International MultiConference of Engineers and Computer Scientists (IMECS), 16–18 March 2011, vol. I, pp. 442–447 (2011)

Prefix and Suffix Sequential Pattern Mining

Rina Singh[✉], Jeffrey A. Graves, Douglas A. Talbert, and William Eberle

Department of Computer Science, Tennessee Technological University, Cookeville, USA
rsingh43@students.tntech.edu

Abstract. Sequential pattern mining is a challenging problem that has received much attention in the past few decades. The mining of large sequential databases can be very time consuming and produces a large number of unrelated patterns that must be evaluated. In this paper, we explore the problems of frequent prefix, prefix-closed, and prefix-maximal pattern mining along with their suffix variants. By constraining the pattern mining task, we are able to reduce the mining time required while obtaining patterns of interest. We introduce notations related to prefix/suffix sequential pattern mining while providing theorems and proofs that are key to our proposed algorithms. We show that the use of projected databases can greatly reduce the time required to mine the complete set of frequent prefix/suffix patterns, prefix/suffix-closed patterns, and prefix/suffix-maximal patterns. Theoretical analysis shows that our approach is better than the current existing approach, and empirical analysis on various datasets is used to support these conclusions.

1 Introduction

For the past several decades, ever since Agrawal and Srikant first published their paper, Mining Sequential Patterns [1], sequential pattern mining has been of broad and current interest. Sequential pattern mining was originally explored for mining information from customer transaction databases. It has been used to predict purchasing behavior [2], applied to next-item prediction problems in recommender systems [3], used to help define guidelines for patient care [4], informed the verification and development of clinical pathways [5], and even guided product placement within supermarkets [6]. Sequential pattern mining has also been used to suggest customer relationship management strategies for small online businesses [7]. However, as transaction databases continue to grow in size and scope, the time required to mine useful patterns continues to increase. Furthermore, actionable retail information is often time sensitive, and while it is possible to obtain patterns in a reasonable (i.e., short) amount of time using many sequential pattern mining techniques, the patterns obtained are often too small or too general to be useful.

While the basic sequential pattern mining task is unconstrained, several constrained variants have been developed. Some of these constraint-based

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 296–311, 2018.
https://doi.org/10.1007/978-3-319-96136-1_24

approaches can help to reduce mining time while allowing a more narrow focus on the patterns obtained. Examples include mining patterns with recency and compactness constraints [8], gap constraints [9–11], relaxation of itemset/transaction constraints [11], and taxonomy based constraints [11].

Despite these constraint variations of sequential pattern mining, most solutions are ill-suited for identifying all sequences that begin or end with an event or events of interest. We are particularly interested in this task, because we are seeking to help improve the selection of cancer treatments through data-driven modeling and simulation of health trajectories. We see frequent sequence mining as a tool that can help with this. In particular, we see the mining of frequent health data sequences that begin or end with specific cancer treatments of interest as potentially informative. Knowing such sequences could allow us to discover probabilistic rules that link sequences that precede particular cancer treatments to subsequent health trajectories.

To advance us toward our goal, this paper focuses on solving this version of constraint-based sequential pattern mining. We provide formal definitions and notation related to mining patterns having a user-defined prefix or suffix. Several theorems are formulated and proven, which provide insight for developing efficient mining techniques for these prefix and suffix patterns. Our newly proposed algorithms are described, and their weaknesses and strengths are compared to the only previously published existing approach for prefix/suffix pattern mining. We have provided theoretical and empirical analysis demonstrating the improvement of our proposed algorithms over the approach proposed by Kaytoue et al., in their work on mining graph sequences [12]. The extraction of cancer-related data is underway but not yet complete, so empirical analysis has been performed on several other real-world datasets in various domains (e.g., click-streams, retail sales, etc.).

2 Sequential Pattern Mining

Closely related to sequential pattern mining is frequent itemset mining. While frequent itemset mining focuses on discovering statistically relevant co-occurring items, sequential pattern mining focuses on identifying patterns among itemsets presented as a sequence. Great effort has been put forth by the data mining community to mine sequential patterns. A number of algorithms have been proposed to mine the complete collection of frequent patterns (i.e., sequences) from a sequential database. Examples include SPADE [13], SPAM [14], and PrefixSpan [15]. Sequential pattern mining can produce large result sets, partially due to the combinatorial nature of the mining task and redundant information represented in the results. To eliminate this redundant information, several algorithms have been proposed to mine frequent closed patterns and frequent maximal patterns. BIDE [16] and CloSpan [17] are examples of such algorithms.

However, as pointed out by Kaytoue et al., none of these algorithms can directly mine patterns given a user-defined prefix or suffix and must be adapted [12]. Prefix and suffix patterns can be useful for uncovering cause and effect relationships. For instance, we are interested in identifying the most common health trajectories occurring after a particular cancer treatment. In this case, the cancer treatment serves as a prefix, for which we want to know the effects.

Fournier–Viger et al., provide an excellent survey on sequential pattern mining [18], and it will be assumed that the reader is familiar with basic definitions, notation, and results from sequential pattern mining. The following section introduces definitions, notation, and results related to prefix/suffix pattern mining used throughout this paper.

3 Prefix/Suffix Pattern Mining

In their research, Kaytoue et al., focus on describing topological changes in dynamic attributed graphs [12]. In order to accomplish this, the authors formally defined the notion of a prefix-closed pattern and a variant on the sequential pattern mining problem they call prefix-closed pattern mining.

3.1 Prefix/Suffix Sequences

Given a fixed sequence A, one may be interested in sequences that "begin" with A or "end" with A. In the former case, A serves as a prefix, and the patterns of interest are called suffix patterns. In the latter case, A serves as a suffix, and the patterns of interest are called prefix patterns. Note that no restrictions are placed on A, and it can be of arbitrary length and could be a sequence of items (i.e., an item sequence) or a sequence of sets of items (i.e., an itemset sequence).

Definition 1. *Let $A = \{a_i\}_{i=1}^{m}$ and $B = \{b_i\}_{i=1}^{n}$ be sequences. Let $C = A \oplus B$ denote the sequence $\{c_i\}_{i=1}^{m+n}$ given by* **juxtaposition** *of A and B, where*

$$c_i = \begin{cases} a_i & 1 \leq i \leq m \\ b_{m-i} & m < i \leq m+n. \end{cases}$$

Definition 2. *Let P and S be non-empty sequences and let $A = P \oplus S$. The sequence P is called a* **prefix** *of A and the sequence S is called a* **suffix** *of A. The sequence A is sometimes called a P-***prefix sequence** *or an S-***suffix sequence***. Alternatively, let $A = \{a_i\}_{i=1}^{m}$ and $B = \{b_i\}_{i=1}^{n}$ be sequences with $m < n$. If $a_i = b_i$ for $1 \leq i \leq m$, then A is called a prefix of B. If $a_i = b_{n-m+i}$ for all $1 \leq i \leq m$, then A is called a suffix of B.*

Note that, unlike in the definition of a subsequence, we require equality of elements (both items and itemsets) in the definition of a prefix/suffix. For example, the sequence $\langle\{a\},\{b\}\rangle$ is a prefix of $\langle\{a\},\{b\},\{c\}\rangle$, but it is not a prefix of $\langle\{a,b\},\{c\}\rangle$, $\langle\{a,x\},\{b,y\},\{c\}\rangle$, or $\langle\{a\},\{x\},\{b\},\{y\}\rangle$.

Definition 3. *A sequence A is said to be* **prefix-closed***, with respect to a database D and suffix B, if B is a suffix of A (that is to say, $A = A' \oplus B$ for some non-empty sequence A') and no proper supersequence of A, also having suffix B, has the same support as A in D. A sequence A is said to be* **prefix-maximal***, with respect to a set S and suffix B, if B is a suffix of A and there exists no proper supersequence of A in S also having suffix B. Analogous definitions exist for* **suffix-closed** *and* **suffix-maximal** *sequences as well.*

Consider the example sequential database presented in Fig. 1. The sequence $\langle\{b\},\{f\},\{e\}\rangle$ is a $\langle\{f\},\{e\}\rangle$-suffix sequence. It is prefix-closed, and has a support of two, but it is not prefix-maximal in the collection of all frequent prefix sequences with positive support as it is a subsequence of $\langle\{a,b\},\{f\},\{e\}\rangle$, which is also a $\langle\{f\},\{e\}\rangle$-suffix sequence of positive support.

Again, consider the sequential database from Fig. 1. The sequence $\langle\{f,g\},\{g\}\rangle$ is a $\langle\{f,g\}\rangle$-prefix sequence with a support of one. It is not suffix-closed as it is a subsequence of $\langle\{f,g\},\{g\},\{e\}\rangle$, which is also $\langle\{f,g\}\rangle$-prefix sequence with the same support; the sequence $\langle\{f,g\},\{g\},\{e\}\rangle$ is both suffix-closed and suffix-maximal (in the set of frequent suffix patterns with positive support).

ID	Sequence
1	$\langle\{a,b\},\{c\},\{f,g\},\{g\},\{e\}\rangle$
2	$\langle\{a,d\},\{c\},\{b\},\{a,b,e,f\}\rangle$
3	$\langle\{a\},\{b\},\{f\},\{e\}\rangle$
4	$\langle\{b\},\{f,g\}\rangle$

Fig. 1. A sample sequential database

It is worth noting that, just like with closed/maximal sequences, every prefix-maximal sequence is prefix-closed, but not every prefix-closed sequence is prefix-maximal, and same thing is true for suffix-closed and suffix-maximal [15,17].

3.2 Prefix/Suffix Projections

Database projections are often used to help reduce the number of database scans and eliminate redundant computation during the mining task [15,17]. In our proposed algorithms, we make use of database projections as a preprocessing step in order to reduce the search space associated with prefix/suffix mining task. This section introduces definitions and notation involving prefix and suffix projections related to our proposed algorithms.

Definition 4. *Let A and B be sequences. The* **prefix-projection** *of A by B (or B-***prefix-projection** *of A) is the longest subsequence A' of A such that $A = B' \oplus A'$ and $B \preceq B'$, which is denoted $\mathcal{P}_B(A) = A'$. If no such subsequence exists, the prefix-projection is defined to be the empty sequence.*

Definition 5. *Let A and B be sequences. The* **suffix-projection** *of A by B (or B-***suffix-projection** *of A) is the longest subsequence A' of A such that $A = A' \oplus B'$ and $B \preceq B'$, which is denoted $\mathcal{S}_B(A) = A'$. If no such subsequence exists, the suffix-projection is defined to be the empty sequence.*

Definition 6. *Let S be a sequence and D be a sequential database. The collection of all S-prefix-projected sequences in D, denoted $\mathcal{P}^*_S(D)$, is called the S-***prefix-projected database** *of D. The collection of all S-suffix-projected sequences in D, denoted $\mathcal{S}^*_S(D)$, is called the S-***suffix-projected database** *of D.*

3.3 Sequential Prefix/Suffix Mining Problems

While the initial goal of Kaytoue et al., was not to advance the area of sequential pattern mining problem, they were the first to propose the prefix-closed mining variant of the sequential pattern mining problem.

Problem 1. Let D be a sequential database, S be a sequence, and n a user-defined minimum support. The ***prefix (closed/maximal) sequential pattern mining problem*** asks to find the complete set of frequent prefix (closed/maximal) patterns with respect to database D and suffix S. The ***suffix (closed/maximal) sequential pattern mining problem*** asks to find the complete set of frequent suffix (closed/maximal) patterns with respect to database D and prefix S.

Kaytoue and his fellow authors claim that the adaption of closed patterns to prefix-closed patterns is not straightforward and proposed a new algorithm. This algorithm is based on a new theorem, developed and proven by Kaytoue and his collogues, which states that the collection of frequent prefix-closed patterns are contained within the collection of closed patterns.

Theorem 1. *For every frequent prefix-closed pattern A, there exists a frequent closed pattern B such that $A \preceq B$ and* $\text{support}(A) = \text{support}(B)$.

In their work on dynamic attributed graphs, Kaytoue and his colleagues only mentioned prefix-closed patterns. However, their observations hold for prefix-maximal patterns and analogous results can be formulated for suffix-closed and suffix-maximal patterns. While this result is interesting from a theoretic viewpoint, we claim that it does not provide a basis for the most efficient algorithm for solving the prefix-closed mining problem. We have proven the following theorem, which provides an alternate method for finding the set of frequent prefix, prefix-closed, and prefix-maximal patterns.

Theorem 2. *Let A be a sequence, n be a user-defined minimum support, and D be a sequential database. Then,*

(i) The complete collection of frequent prefix patterns with respect to database D having suffix A and minimum support n is given by $\{P = P' \oplus A \mid P'$ is frequent in $\mathcal{S}^*_A(D)\}$.

(ii) The complete collection of prefix-closed patterns with respect to database D having suffix A and minimum support n is given by $\{P = P' \oplus A \mid P'$ is closed in $\mathcal{S}_A^*(D)\}$.
(iii) The complete collection of prefix-maximal patterns with respect to database D having suffix A and minimum support n is given by $\{P = P' \oplus A \mid P'$ is maximal in $\mathcal{S}_A^*(D)\}$.

Analogous results for the suffix variation of Theorem 2 exists but have been omitted. Part (i) in Theorem 2 is not surprising and can be established with a fairly straightforward direct proof. What is not so obvious are Parts (ii) and (iii). Let A be a sequence, n a minimum support, and D a sequential database. If P' is a closed/maximal pattern in $\mathcal{S}_A^*(D)$, why must the prefix pattern $P = P' \oplus A$ also be prefix-closed/maximal in D? If $P = P' \oplus A$ is prefix-closed/maximal pattern in D, why must P' also be closed/maximal in $\mathcal{S}_A^*(D)$?

Proof. The proof of Part (iii) is analogous to that of Part (ii). To establish (ii), we need only to show that every prefix-closed pattern having suffix A is of the from $P = P' \oplus A$ for some sequence P' which is closed in $\mathcal{S}_A^*(D)$, and that every sequence of the from $P = P' \oplus A$ where P' is closed in $\mathcal{S}_A^*(D)$ is a prefix-closed pattern in D.

($\subseteq$) Suppose that P is a prefix-closed pattern with respect to suffix A in database D. Then $P = P' \oplus A$ for some non-empty sequence P'. We proceed by way of contradiction. Suppose that P' is not closed in $\mathcal{S}_A^*(D)$. Then, there exists a supersequence, say P'', such that $P' \prec P''$ and $\text{support}_{\mathcal{S}_A^*(D)}(P') = \text{support}_{\mathcal{S}_A^*(D)}(P'')$. Put $S = P'' \oplus A$, and observe that $P \prec S$. As established in the proof of Theorem 2, we have that $\text{support}_D(P) = \text{support}_{\mathcal{S}_A^*(D)}(P')$ and $\text{support}_D(S) = \text{support}_{\mathcal{S}_A^*(D)}(P'')$. It follows that $\text{support}_D(P) = \text{support}_D(S)$, a contradiction to the assumption that P was closed.

($\supseteq$) Now, suppose that $P = P' \oplus A$ for some closed sequence P' in $\mathcal{S}_A^*(D)$. We want to show that P is prefix-closed. Suppose not. That is, suppose that there exists an A-suffix sequence, say S, such that $P \prec S$ and $\text{support}_D(P) = \text{support}_D(S)$. Since S and $P = P' \oplus A$ are both A-suffix sequences, we have that $S = P'' \oplus A$ for some sequence $P'' \prec P'$. Since $\text{support}_D(P) = \text{support}_{\mathcal{S}_A^*}(P')$, $\text{support}_D(S) = \text{support}_{\mathcal{S}_A^*}(P'')$, and $\text{support}_D(P) = \text{support}_D(S)$, it is the case that $\text{support}_{\mathcal{S}_A^*}(P') = \text{support}_{\mathcal{S}_A^*}(P'')$. That is to say, P' is not closed, a contradiction.

3.4 Algorithms

Upon first thought, one may think that there should be a relationship between the complete set of frequent prefix patterns, prefix-closed patterns, and prefix-maximal patterns and the complete set of frequent patterns, closed patterns, and maximal patterns, respectively. Theorem 1 provided by Kaytoue et al., proves this to be true [12]. To mine the complete set of frequent prefix-closed patterns with respect to a database D and a suffix S, begin by mining the complete set of frequent closed patterns in D. Then for each pattern A, consider the S-suffix-projected sequence $\mathcal{S}_S(A) = A'$ of A. If A' is non-empty, the sequence $A' \oplus S$ is a potential prefix-closed sequences. Some of the sequences obtained from this suffix projection-extension process may not be prefix-closed. After filtering out the non-prefix-closed patterns, the complete set of frequent prefix-closed patterns will be all that remains. Kaytoue et al., use a subsumption checking algorithm to perform this filtering. This idea is summarized in Algorithm 1; analogous algorithms exist for prefix pattern mining, prefix-maximal pattern mining, and their suffix mining variants.

Algorithm 1. Kaytoue Based Prefix-Closed Pattern Mining

```
Require: $D$: a sequential database
Require: $S$: a suffix for prefix mining
Require: $n$: a user-defined minimum support
 1: procedure ClosePrefixMiner($D$, $S$, $n$)
 2:     $C \leftarrow$ ClosedPatternMining($D$, $n$)
 3:     $C' \leftarrow [\,]$
 4:     for $A' \in C$ do
 5:         if $\mathcal{S}_S(A') \neq \langle\rangle$ then
 6:             $A \leftarrow \mathcal{S}_S(A') \oplus S$
 7:             $C' \leftarrow C' + [A]$
 8:         end if
 9:     end for
10:     $C' \leftarrow$ SubsumptionFiltering($C'$)
11:     return $C'$
12: end procedure
```

Theorem 2 provides an alternate approach for obtaining the complete set of frequent prefix, prefix-closed, and prefix-maximal patterns with an analogous version existing for their suffix counterparts. It is fairly easy to see that the complete set of frequent prefix patterns can be obtained from a suffix-projected database. What is not so obvious is that the complete set of prefix-closed and prefix-maximal patterns (along with their suffix variants) corresponds directly to the complete set of prefix-closed and prefix-maximal patterns found in the associated suffix-projected database. For example, to obtain the complete set of prefix-closed patterns from a database D having suffix S, begin by constructing the suffix-projected database $\mathcal{S}^*_S(D)$. Then, mine the complete set of closed patterns from $\mathcal{S}^*_S(D)$. For each closed pattern P obtained, the sequence $P \oplus S$ is guaranteed to be prefix-closed, by Theorem 2. This idea is summarized in Algorithm 2; analogous algorithms exist for prefix pattern mining, prefix-maximal pattern mining, and their suffix mining variants. In Sect. 4, we will demonstrate the advantage of Algorithm 2 over Algorithm 1.

If one is interested in obtaining prefix-closed patterns with respect to multiple suffixes, these algorithms can be modified in order to obtain the desired results. In the case of Algorithm 1, we need only to mine the complete set of closed patterns once. Then, each of the suffixes can be used in turn to extract the complete set of prefix-closed patterns, as shown in Algorithm 3. The extension to Algorithm 2 is a little more complicated. For each of the desired suffixes, a suffix-projected database must first be constructed. Then the complete set of closed patterns can be mined (see Algorithm 4). Having to mine multiple projected databases for closed patterns may seem inefficient, but as we will demonstrate in Sect. 4, this is actually faster.

4 Theoretical Analysis

In this section, we will illustrate the advantages of mining projected databases (i.e., Algorithms 2 and 4) over that of the full sequential database (i.e., Algorithms 1 and 3). We should first point out, that while our approach is in the same computational complexity class as Kaytoue's approach, it can result in drastically shorter runtime. This can be attributed to a reduced search space or a reduced input problem size, depending on the point of view.

Algorithm 2. Efficient Prefix-Closed Pattern Mining

```
Require: $D$: a sequential database
Require: $S$: a suffix for prefix mining
Require: $n$: a user-defined minimum support
 1: procedure ClosePrefixMiner($D$, $S$, $n$)
 2:     $D' \leftarrow \mathcal{S}_S^*(D)$
 3:     $C \leftarrow$ ClosedPatternMining($D'$, $n$)
 4:     $C' \leftarrow [\,]$
 5:     for $A' \in$ C do
 6:         $A \leftarrow A' \oplus S$
 7:         $C' \leftarrow C' + [A]$
 8:     end for
 9:     return $C'$
10: end procedure
```

Algorithm 3. Kaytoue Based Prefix-Closed Pattern Mining w/ Multiple Suffixes

```
Require: $D$: a sequential database
Require: $L_S$: a list of suffixes for prefix mining
Require: $n$: a user-defined minimum support
 1: procedure ClosePrefixMiner++($D$, $L_S$, $n$)
 2:     $C \leftarrow$ ClosedPatternMining($D$, $n$)
 3:     $C' \leftarrow [\,]$
 4:     for $S \in L_S$ do
 5:         $C'_S \leftarrow [\,]$
 6:         for $A' \in C$ do
 7:             if $\mathcal{S}_S(A') \neq \langle\rangle$ then
 8:                 $A \leftarrow \mathcal{S}_S(A') \oplus S$
 9:                 $C'_S \leftarrow C'_S + [A]$
10:             end if
11:         end for
12:         $C'_S \leftarrow$ SubsumptionFiltering($C'_S$)
13:         $C' \leftarrow C' + C'_S$
14:     end for
15:     return $C'$
16: end procedure
```

Algorithm 4. Efficient Prefix-Closed Pattern Mining w/ Multiple Suffixes

```
Require: $D$: a sequential database
Require: $L_S$: a list of suffixes for prefix mining
Require: $n$: a user-defined minimum support
1: procedure ClosePrefixMiner++($D$, $L_S$, $n$)
2:     $C' \leftarrow []$
3:     for $S \in L_S$ do
4:         $C'_S \leftarrow$ ClosePrefixMiner($D, S, n$)
5:         $C' \leftarrow C' + C'_S$
6:     end for
7:     return $C'$
8: end procedure
```

4.1 Reduced Problem Size

It is obvious that Algorithm 2 has a smaller database to mine than Algorithm 1; the smaller the projected-database is compared to the original database, the faster the closed-pattern mining task will be. First, note that the creation of the projected database is linear in terms of the number of sequences in the database, the length of the sequences in the database, and the length of the suffix in question. On the other hand, the mining task is exponential in terms of the length of the sequences being mined. To see this, let $\mathcal{I}$ be a collection of items. The number of non-empty item sequences of length at most k, for some positive constant k, is given by

$$\sum_{i=1}^{k} |\mathcal{I}|^k = \frac{|\mathcal{I}|^{k+1} - |\mathcal{I}|}{|\mathcal{I}| - 1}.$$

In the case of itemset sequences, the number of non-empty subsets of $\mathcal{I}$ is $2^{|\mathcal{I}|} - 1$, and so the number of itemset sequences of length at most k is given by

$$\sum_{i=1}^{k} (2^{|\mathcal{I}|} - 1)^k = \frac{(2^{|\mathcal{I}|} - 1)^{k+1} - (2^{|\mathcal{I}|} - 1)}{(2^{|\mathcal{I}|} - 1) - 1}.$$

Hence, for a fixed itemset $\mathcal{I}$, the number of sequences of length at most k grows exponentially with k.

In the case of sequential pattern mining, the value of k is determined by the length of the sequences in the database. More specifically, if the number of sequences in the database of length at least k is fewer then a user-defined minimum absolute support m, then no sequence of length k will be frequent. Let f_D denote the function that counts the number of sequences in a database D of length at least n, which is given by $f_D(n) = |\{S \in D \mid |S| \geq n\}|$. Then $\max\{n \mid f_D(n) \geq m\}$, where m is a user-defined minimum absolute support, places an upper bound on the value of k. And so, Algorithm 2 performs a linear number of computations in order to reduce the input size of the exponential closed pattern mining task.

The length of the suffix and its support within the original database will affect the size of the projected database. The smallest reduction in database size occurs when the suffix used in prefix-closed mining is a suffix of every sequence in the database. In this case, every one of the sequences in the suffix-projected database is smaller than their corresponding sequence in the original database by precisely the size of the suffix used to obtain the projections. We expect the smallest possible speedup of Algorithm 2 over Algorithm 1 to occur when the suffix of interest is of length one. In practice, Algorithm 2 may be slower due to the overhead of constructing the sequential database.

Conversely, the largest reduction in database size will occur when the the suffix in question has low support in the given database. In this situation, many suffix-projected sequences within the suffix-projected database will be empty. In the extreme, the suffix will have zero support within the database, and the closed pattern mining step in Algorithm 2 will return almost immediately. This will not be the case for Algorithm 1, which will proceed to mine the complete set of closed patterns, none of which will contain the suffix of interest. In this case, the largest speedup should occur.

4.2 Reduced Search Space

An alternative explanation for the improved running time of Algorithm 2 over Algorithm 1 is the smaller search space explored by the former algorithm. Figure 2 depicts the standard search tree used to generate all item sequences given an itemset $\mathcal{I} = \{a, b, c\}$; a similar tree exists for generating all itemset sequences. Many frequent sequential pattern mining algorithms implicitly search this tree of all sequential patterns using various techniques for pruning. Consider instead the frequent suffix-closed mining variant of Algorithm 2. This algorithm will produce the complete set of frequent suffix-closed patterns, given a database D, prefix P, and minimum support n. By first constructing a P-prefix projected database, this new algorithm can be thought of as exploring a subspace of the complete search tree seen in Fig. 2. For example, given a prefix of $\langle b, a\rangle$, only the ba subtree shaded in Fig. 2 needs to be explored.

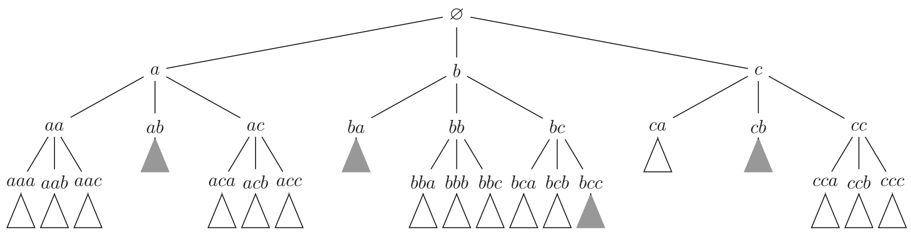


Fig. 2. Standard sequence space with $\mathcal{I} = \{a, b, c\}$.

4.3 Mining Several Prefix/Suffix Patterns

On the surface, Algorithm 4 may appear worse than Algorithm 3 for mining prefix-closed patterns when multiple suffixes of interest are given. Obviously it must solve several instances of the closed pattern mining problem while Algorithm 3 only has to solve one instance. Again, we will explore the dual frequent suffix-closed pattern mining algorithm to see that the use of projected sequences is still beneficial.

Given several prefixes of interest, the suffix-closed variant of Algorithm 4 can be thought to explore several subtrees within the complete sequence search tree. So long as none of the given prefixes happens to be a prefix of another, these subtrees are disjoint within the search space of all sequences. For example, given an $\mathcal{I} = \{a, b, c\}$ with suffixes $\langle a, b\rangle$, $\langle b, a\rangle$, $\langle c, b\rangle$, and $\langle b, c, c\rangle$ of interest, only the *ab*, *ba*, *cb*, *bcc* subtrees shaded in Fig. 2 need to be explored. As the number of prefixes for use in suffix-closed mining increase, larger portions of the search tree must be explored; if every possible item is included as a prefix sequence of length one, then the entire space must be explored. In this case, Algorithm 4 should not be expected to provided any speedup over Algorithm 3, and may be slower due to the overhead of building the projected databases.

5 Empirical Analysis

In order to show that our proposed algorithms are useful in practice, we have implemented Algorithms 1 to 4 using C++. It is worth noting that, for ease of implementation, physical prefix-projected and suffix-projected databases were created for use by the sequential pattern miner in Algorithms 2 and 4 (pseudo-projections were used within the actual sequential pattern miners). This allows for easy replacement of the frequent/closed/maximal mining algorithm based on dataset characteristics. This is desirable as some algorithms work better on long sequences with few possible items while others work better on short sequences with many possible items [18].

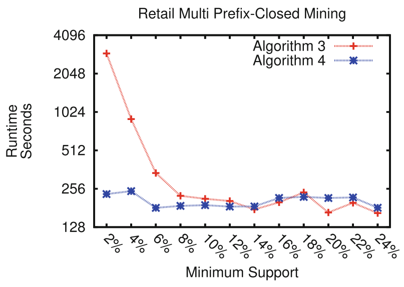
However, the use of physical projected databases will result in a larger memory footprint and additional overhead when creating the physical projection. The use of a pseudo-projected database would eliminate much of the overhead associated with creating the prefix-projected and suffix-projected databases. In addition, it would require no more memory than that of Algorithms 1 and 3 because of the smaller search space examined. The downside is that some sequential pattern minings would require modification if a pseudo-projection is used. For example, the ***backward-extension checking*** step used in the BIDE closed-pattern mining algorithm [16] would eliminate a suffix-closed pattern, with prefix P, of the form $P \oplus A$ if there exists a supersequence $P' \oplus A$, where $P \prec P'$, having the same support; this is a problem since P' would not have prefix P.

5.1 Empirical Setup

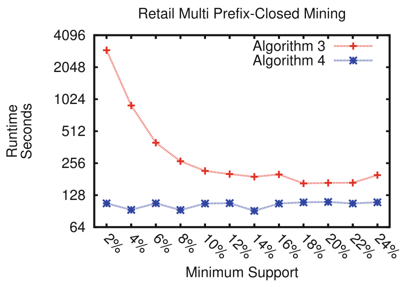
We have tested our algorithm on several sequential databases from various domains, with datasets taken primarily from Fournier-Viger's SPMF website [19]. While there are several constraint-based sequential pattern mining problems, Kaytoue's proposed approach is the only one that attempts to solve the prefix/suffixed closed pattern mining problem, and so we use it as abaseline for comparison (i.e., Algorithms 1 and 3 against Algorithms 2 and 4). Experiments were performed on DELL c6320 servers. Each machine contained dual Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40 GHz (28 physical cores) and 128 GB of RAM; the algorithms implemented were serial and could not take advantage of multiple cores. We chose to focus on the prefix mining variant simply because these are of most interest to us. The suffix based variants produced similar results and have been omitted for brevity. In addition, the results presented will focus on prefix-patterns where the suffix of interest is a sequence consisting of a single item. This decision was made as it addresses the most difficult prefix-closed mining task; mining sequences of itemsets or allowing for suffixes of length greater than one affords more opportunities for search space pruning, and we wish to focuses on worst-case situations for our proposed algorithms.

5.2 Retail

The Retail dataset consists of transactions from a UK-based and registered non-store online retail [20]. Timestamp information along with invoice ids and customer ids were used to build sequences, resulting in 4,372 sequences with 3,684 items. Each sequence represents a customer, while the items represent unique all-occasion gifts sold by the retail. Since multiple items can appear on an invoice, this sequential database consists of itemset sequences, with the largest itemset consisting of 541 items. Figure 3 illustrates the time required to mine all prefix-closed patterns using the top five most and least supported suffixes. These plots show that, when the mining task is difficult (i.e., the minimum support is low or the support of the suffix is low), Algorithm 4 significantly outperforms Algorithm 3. This is due to the fact that the mining time greatly exceeds the time required to build the projected databases. Conversely, if the mining task is easy, Algorithm 4 may perform worse than Algorithm 3 due to the overhead associated with building the projected databases.



(a) Five Most Frequent Suffixes

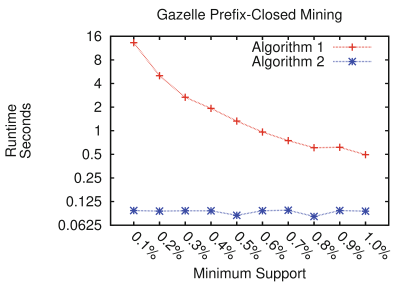


(b) Five Least Frequent Suffixes

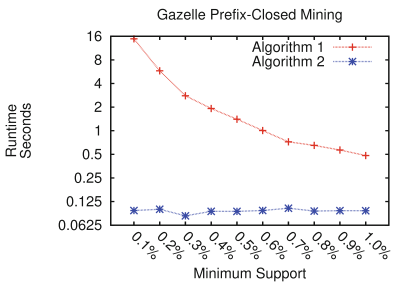
Fig. 3. Retail dataset

5.3 Click-Streams

The Gazelle dataset (i.e., the KDD CUP 2000 dataset) contains click-stream data relating to purchases from a web retailer. It contains of 77,512 sequences consisting of 3,340 unique items, with the most frequent item having 4.86% support. We selected the most and least frequently occurring items to serve as suffixes to test Algorithms 1 and 2. The runtime results can be seen in Fig. 4, which shows that Algorithm 2 outperforms Algorithm 1.



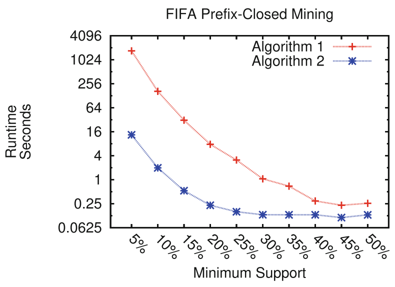
(a) Most Frequent Suffix (4.86%)



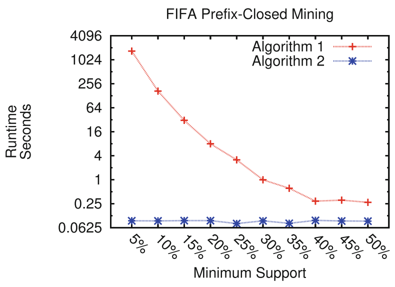
(b) Least Frequent Suffix (0.0013%)

Fig. 4. Gazelle dataset

The FIFA dataset consists of 20,450 sequences made up of 2,990 items representing click-streams gathered from the FIFA World Cup 98 website. The top ten most frequently viewed webpages all had over 35% support within the database. There were several pages that were only viewed a single time resulting in a support less than 0.005%. Figure 5 depicts the runtimes for mining the single item suffix with the highest and lowest support. The most frequently occurring item within the FIFA dataset poses more difficulty for mining, but Algorithm 2 still outperforms Algorithm 1.



(a) Most Frequent Suffix (42.91%)

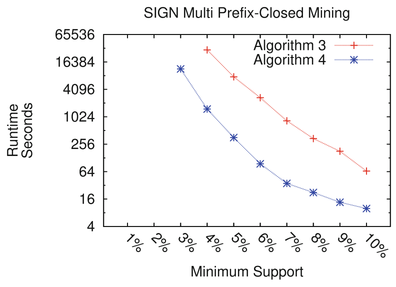


(b) Least Frequent Suffix (0.0049%)

Fig. 5. FIFA dataset

5.4 Natural Languages

The SIGN dataset consists of 730 sequences with 267 unique symbols, and this dataset also contained some of the longest sequences found among all datasets we explored. The dataset was created by the National Center for Sign Language and Gesture Resources at Boston University. Each sequences represents an "utterance" consisting of ASL gestures and facial expressions [21]. Due to the length of the sequences and relatively few symbols, the SIGN dataset required the largest amount of time to extract prefix-closed patterns. The result seen in Fig. 6 shows the time required to mine the complete set of prefix-closed patterns when every possible single item suffix is used. Despite an identical search space explored by both algorithms, Algorithm 4 still outperforms Algorithm 3. This is most likely because the subsumption checking used in Algorithm 3 is more computationally expensive than projected database creation used in Algorithm 4.



All Possible Single Item Suffixes

Fig. 6. SIGN dataset

6 Conclusions and Future Work

In this paper, we introduced the problems of frequent prefix mining, prefix-closed mining, and prefix-maximal mining along with their suffix variants. We have shown the usefulness of mining projected databases for obtaining prefix/suffix patterns and have proven that these approaches produce the complete set of frequent prefix/suffix patterns. Theoretical analysis shows that it is better to create multiple projected databases when faced with multiple prefixes/suffixes of interest (as apposed to mining the original database a single time), and empirical analysis supports this conclusion. Empirical analysis also shows that our proposed algorithms, while not more efficient in the sense of being in a better complexity class, tend to be an order of magnitude faster in practice.

In the future, we want to explore the use of prefix/suffix sequential pattern mining for predicting health trajectories of cancer treatments. By mining suffix patterns related to cancer treatments, we hope to develop probabilistic rules that link particular cancer treatments to subsequent health trajectories. In addition, we want to leverage prefix pattern mining in the hopes that prior medical history will allow us to better predict health trajectories after a particular treatment.

References

1. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the Eleventh International Conference on Data Engineering (1995)
2. Huang, C.L., Huang, W.L.: Handling sequential pattern decay: developing a two-stage collaborative recommender system. Electron. Commer. Res. Appl. **8**(3), 117–129 (2009)
3. Yap, G.-E., Li, X.-L., Yu, P.S.: Effective next-items recommendation via personalized sequential pattern mining. In: Lee, S., et al. (eds.) DASFAA 2012. LNCS, vol. 7239, pp. 48–64. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29035-0_4
4. Baralis, E., et al.: Analysis of medical pathways by means of frequent closed sequences. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010. LNCS (LNAI), vol. 6278, pp. 418–425. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15393-8_47
5. Uragaki, K., Hosaka, T., Arahori, Y., Kushima, M., Yamazaki, T., Araki, K., Yokota, H.: Sequential pattern mining on electronic medical records with handling time intervals and the efficacy of medicines. In: 2016 IEEE Symposium on Computers and Communication (ISCC) (2016)
6. Aloysius, G., Binu, D.: An approach to products placement in supermarkets using PrefixSpan algorithm. J. King Saud Univ.-Comput. Inf. Sci. **25**(1), 77–87 (2013)
7. Shim, B., Choi, K., Suh, Y.: CRM strategies for a small-sized online shopping mall based on association rules and sequential patterns. Expert Syst. Appl. **39**(9), 7736–7742 (2012)
8. Chen, Y.L., Hu, Y.H.: Constraint-based sequential pattern mining: the consideration of recency and compactness. Dec. Support Syst. **42**(2), 1203–1215 (2006)
9. Antunes, C., Oliveira, A.L.: Generalization of pattern-growth methods for sequential pattern mining with gap constraints. In: Perner, P., Rosenfeld, A. (eds.) MLDM 2003. LNCS, vol. 2734, pp. 239–251. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45065-3_21
10. Li, C., Wang, J.: Efficiently mining closed subsequences with gap constraints. In: Proceedings of the 2008 SIAM International Conference on Data Mining (2008)
11. Srikant, R., Agrawal, R.: Mining sequential patterns: generalizations and performance improvements. In: Apers, P., Bouzeghoub, M., Gardarin, G. (eds.) EDBT 1996. LNCS, vol. 1057, pp. 1–17. Springer, Heidelberg (1996). https://doi.org/10.1007/BFb0014140
12. Kaytoue, M., Pitarch, Y., Plantevit, M., Robardet, C.: What effects topological changes in dynamic graphs? Elucidating relationships between vertex attributes and the graph structure. Soc. Netw. Anal. Min. **5**, 55 (2015)
13. Zaki, M.J.: SPADE: an efficient algorithm for mining frequent sequences. Mach. Learn. **42**(1–2), 31–60 (2001)
14. Ayres, J., Flannick, J., Gehrke, J., Yiu, T.: Sequential pattern mining using a bitmap representation. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2002)

15. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.C.: PrefixSpan: mining sequential patterns efficiently by prefix-projected pattern growth. In: Proceedings of the 17th International Conference on Data Engineering (2001)
16. Wang, J., Han, J.: BIDE: efficient mining of frequent closed sequences. In: Proceedings of the 20th International Conference on Data Engineering (2004)
17. Yan, X., Han, J., Afshar, R.: CloSpan: mining closed sequential patterns in large datasets. In: Proceedings of the 2003 SIAM International Conference on Data Mining (2003)
18. Fournier-Viger, P., Lin, J.C.W., Kiran, R.U., Koh, Y.S., Thomas, R.: A survey of sequential pattern mining. Data Sci. Pattern Recogn. **1**(1), 54–77 (2017)
19. Fournier-Viger, P., et al.: The SPMF open-source data mining library version 2. In: Berendt, B., et al. (eds.) ECML PKDD 2016. LNCS (LNAI), vol. 9853, pp. 36–40. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46131-1_8
20. Chen, D., Sain, S.L., Guo, K.: Data mining for the online retail industry: a case study of RFM model-based customer segmentation using data mining. J. Database Mark. Cust. Strategy Manag. **19**(3), 197–208 (2012)
21. Neidle, C.: SignStream^{TM}: a database tool for research on visual-gestural language. Sign Lang. Linguist. **4**, 203–214 (2001)

Long Short-Term Memory Recurrent Neural Network for Stroke Prediction

Pattanapong Chantamit-o-pas[1,2](✉) and Madhu Goyal[1]

[1] Centre for Artificial Intelligence,
Faculty of Engineering and Information Technology,
University of Technology Sydney,
PO Box 123 Broadway, Ultimo, NSW 2007, Australia
Pattanapong.Chantamit-o-pas@student.uts.edu.au,
Madhu.Goyal-2@uts.edu.au

[2] Faculty of Information Technology,
King Mongkut's Institute of Technology Ladkrabang,
Ladkrabang, Bangkok 10520, Thailand
Pattanapong@it.kmitl.ac.th

Abstract. Electronic Healthcare Records (EHRs) describe the details about a patient's physical and mental health, diagnosis, lab results, treatments or patient care plan and so forth. Currently, the International Classification of Diseases, 10th Revision or ICD-10 code is used for representing each patient record. The huge amount of information in these records provides insights about the diagnosis and prediction of various diseases. Various data mining techniques are used for the analysis of data deriving from these patient records. Recurrent Neural Network (RNN) is a powerful and widely used technique in machine learning and bioinformatics. This research aims at the investigation of RNN with Long Short-Term Memory (LSTM) hidden units. The empirical research is intended to evaluate the ability of LSTMs to recognize patterns in multi-label classification of cerebrovascular symptoms or stroke. First, we integrated ICD-10 code into health record, as well as other potential risk factors within EHRs into the pattern and model for prediction. Next, we modelled the effectiveness of LSTMs for prediction of stroke based on healthcare records. The results show several strong baselines that include accuracy, recall, and F1 measure score.

Keywords: Deep learning · Cerebrovascular disease · Predictive technique LSTM-RNN

1 Introduction

Deep learning algorithm is a technique that focuses on how computers learn from data. It is the intersection of statistics, computer science, and mathematics - which generates the algorithm of building patterns and models from massive data sets, as well as is applicable to billons or trillions of data records [1, 2]. Deep learning technique employs learning from data together with multiple levels of abstraction deriving from computational models that are associated with multiple processing layers.

Basically, a cerebrovascular disease or stroke is the state of lacking blood supply in an area of the brain, and it happens once a vessel is blocked. This is known as an "*ischemic stroke*", where about three-quarters of vessels are blocked. Meanwhile, a "*hemorrhagic stroke*" refers to the state where a blood vessel bursts. It can also affect different parts of the human body, depending on which area of brain is affected. In most countries, stroke becomes the second or third common cause of death [3, 4]. The patients who survived usually have poor quality of life because of serious illness, long-term disability and become burden to their families and health care system. This strongly demands for the management focusing on prevention and early treatment of diseases by analysing different factors. According to the analysis, it was found that several health conditions and lifestyle factors become risk factors for stroke.

The predictive techniques of stroke vary from simple to more complex models. The risk factors of stroke are complex and applicable to find different convolutions of disease and uncertainty from direct and/or indirect sources. The analysis of stroke patients who were admitted in the TOAST study was done by using stepwise regression methods [5]. This research was conducted among 1,266 stroke patients selected from database, provided that those patients must have had suffered a transient ischemic attack (TIA) or recurrent stroke within 3 months after the first stroke. Additionally, 20 clinical variables were chosen for finding performance and evaluation.

Some researches show that the use of ICD-9 codes in combination with other health care data can accurately diagnose patients' health issues [6–8]. Most of Electronic Healthcare Records (EHRs) adopt the codes from International Classification of Diseases (ICD), 10th Revision i.e. ICD-10 and ICD-10-CM codes, and those codes become the standard codification in the Electronic Medical Record system (EMR) [9]. The use of International Classification of Diseases in various health care institutions provides the similar basic schemes that allow patient data to be used in a similar way.

The rest of this paper is organized as follows. Section 2 describes related work on deep learning. Section 3 ICD-10 complaint Electronic Healthcare Records. Section 4 describes prediction of stroke using EHRs and deep learning. Section 5 presents the evaluation model. Section 6 discusses the result of this research and the conclusion and future work are present in Sect. 7 of this paper.

2 Deep Learning

Deep Learning method is intended to discover complex structure in big data set by using the advanced mathematical algorithm to predict the result. The machine can learn from source and change its internal parameters by computing the representation in each layer to form the representation in the previous layer.

Basically, deep net has various techniques to predict the result. It is recommended to use either Restricted Boltzmann Machine (RBM) or auto encoder for unsupervised learning and the extraction of pattern from a set of unlabelled data. Several options are usable if there are labelled data for supervised learning, and once it is required to build classifier, depending on specific application. A recurrent network or Recursive Neural Tensor Network (RNTN) can be applied to text processing task like a sentence analysis based on phrasing, name and recognition. Moreover, Deep Belief Network (DBN) or convolutional networks are used for image recognition. The RNTN or convolutional networks are also used for object recognition. Finally, a recurrent network is used for the speed recognition as well. In general, both DBN and Multilayer perceptron – also known as Rectified Linear Units (ReLUs) – are good choices for classification. Also, a recurrent network is the best option for time series analysis.

Gulshan et al. [10] invented an algorithm for automated detection of diabetic retinopathy in Retinal fundus photographs (RDR). In this research, a deep convolutional neural network was employed to optimize image classification and was trained by using a retrospective development dataset of 128,175 images. The results show that an algorithm has high sensitivity and specificity for detecting referable diabetic retinopathy.

In acute ischemic stroke treatment, the prediction of tissue survival outcome plays a fundamental role in the clinical decision-making process as it can be used to assess the balance of risk and possible benefit when endovascular c1otretrieval intervention is investigated. For the first time, Stier et al. [11] constructed a deep learning model of tissue fate based on randomly sampled local patches from the hypoperfusion (Tmax) feature observed in MRI immediately after symptom onset. They evaluated the model with respect to the ground truth established by an expert neurologist four days after intervention. The results show the superiority of the proposed regional learning framework over a single-voxel-based regression model. The previous researches reveal that the kernel of the deep learning techniques can be applied to healthcare sector as a regulariser at the output layers or a part of model.

The conventional models are incapable of detecting fundamental knowledge because they fail to simulate the complexity and feature representation of medical problem domains. Researchers attempt to apply a deep model to overcome this weakness. Several applications of deep learning model to medical data analysis have been reported in recent years, for instance, an image analysis system for histopathological diagnosis on the images. Liang et al. [12] suggested the application of deep belief network for unsupervised feature extraction, and then conducting supervised learning through a standard SVM. The results confirm the advantage of deep model towards knowledge modelling for data from medical information systems such as Electronic Medical Record (EMR) and Hospital Information System (HIS). Thus, predictive analytical techniques for stroke using deep learning techniques are potentially significant and beneficial.

In healthcare fields, data in EHRs are quite significant for decision-making in treatment. In general, a realistic dataset contains useful records for clinical practice. It uncovered realistic environments for the analyses of diseases because it had included ambiguous and incomplete values that contribute to errors and are unsuitable for annalistic data and a very challenging analysis. Normally, it needs to be fulfilled before being used. Hammerla et al. [13] proposed an assessment system that managed practical usability constraints and applied deep learning technique to differentiating disease state in datasets that are naturalistic settings. In this research, a large dataset was collected from 34 participants who suffered Parkinson's Disease (PD).

In other fields, deep learning is used in stock price prediction by extracting structure event from news text. The prediction technique uses event-driven approach. First, the system extracts the events from text and demonstrates dense vectors. Then, it trains using a novel neural tensor network. Second, both short-term and long-term influences of event on stock price moments are combined and processed by using deep convolutional neural network. In comparison with state-of-the-art baseline methods, the results show that this model can achieve approximately 6% improvements on S&P 500 index prediction and individual stock prediction, respectively. In addition, market simulation results show that the system is more capable of making profits than previously reported systems trained on S&P 500 stock historical data [14, 15].

3 ICD-10 Complaint Electronic Healthcare Records

There is significant growth in the amount of medical or patient data being generated in hospitals or clinics all over the world. In most cases, Electronic Health Records are used for storing most of this medical or patient data. In practice, International Classification of Diseases (ICD) are used for electronic health records and for classifying diseases and other health problems appearing in many types of health and vital records. Currently, ICD-10 codes are used by hospitals and health professionals, which are retrieved from Electronic Medical Records (EMR) system. The standard provides a very convenient platform for primary and secondary data analysis of these records for diagnosis and prediction of diseases, as well as for the improvement of medical and patient care. The 'core' three character code of classification of ICD-10 is the mandatory level of coding for international reporting. It also has four character sub-categories which are not mandatory for international reporting [9]. The Electronic Healthcare Records (EHR) of cerebrovascular disease patients contain various information, including demographic data, potential risk factors, and non-potential risk factors that are recorded in hospital database (See Fig. 1).

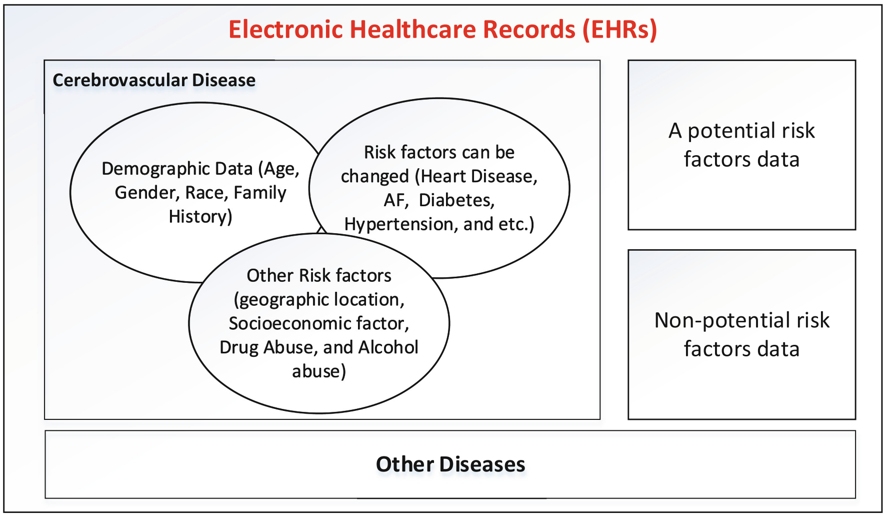

Electronic Healthcare Records (EHRs)
Cerebrovascular Disease
Demographic Data (Age, Gender, Race, Family History)
Risk factors can be changed (Heart Disease, AF, Diabetes, Hypertension, and etc.)
Other Risk factors (geographic location, Socioeconomic factor, Drug Abuse, and Alcohol abuse)
A potential risk factors data
Non-potential risk factors data
Other Diseases

Fig. 1. Electronic healthcare records of stroke patients.

In detail, EHR record consists of gender; data of birth (DOB); clinic operation (CLINIC_OPD); date operation (DATEOPD); date diagnosis (DATEDX); clinic diagnosis (CLINIC_ODX); diagnosis code (DIAG); and diagnosis type (DXTYPE). Gender was identified and represented by either the code 1 (Man) or the code 2 (Woman) (see Table 1). DOB field record contains patient's birthdate and some records have null value or error in term of date. We eliminated the null value or error and converted the value into age in preparation process. CLINIC_OPD field indicates the clinic number of hospital where patient has treatment. DATEOPD field demonstrates the data of service. DATEDX field indicates the date of diagnosis. CLINIC_ODX field shares similar code with CLINIC_OPD field. Code of diagnoses were obtained from doctors or medical experts who used ICD-10 codes and inputted in DIAG field. DXTYPE field specifies types of disease. It consists of (1) primary disease, (2) co-morbidity disease, (3) complication, and (4) other diseases (see Table 1). Therefore, the demographic data, disease type, and other information were recorded once each patient visited. For prediction process, we integrated the multiple value dependencies into EHRs. Further details will be described in Sect. 4.

Table 1. Sample partial electronic healthcare records in hospital database.

Gender	DOB	CLINIC_OPD	DATEOPD	DATEDX	CLINIC_ODX	DIAG	DXTYPE
1	xx/xx/xxxx	120	xx/xx/xxxx	xx/xx/xxxx	120	C20	1
2	xx/xx/xxxx	20	xx/xx/xxxx	xx/xx/xxxx	20	D379	1
2	xx/xx/xxxx	10	xx/xx/xxxx	xx/xx/xxxx	10	C499	1
2	xx/xx/xxxx	120	xx/xx/xxxx	xx/xx/xxxx	120	K297	1
1	xx/xx/xxxx	10	xx/xx/xxxx	xx/xx/xxxx	10	C499	1
2	xx/xx/xxxx	10	xx/xx/xxxx	xx/xx/xxxx	10	I158	2
2	xx/xx/xxxx	120	xx/xx/xxxx	xx/xx/xxxx	120	I694	2

4 Prediction of Stroke Using EHRs and Deep Learning

In this paper, deep learning algorithm is applied on EHRs for prediction of stroke. Deep Learning (DL) is a process of training a neural network to perform given task. Overall the prediction of Stroke has two main steps i.e. selection of EHRs based on risk factors and prediction process. In the first step, first the null values and anomaly data were eliminated via JAVA programming. Then the ICD-10 codes were filtered by stroke's risk factors. The EHRs with ICD-10 codes are then filtered again to eliminate anomaly data such as negative values, null values etc. Then, EHRs files consisting of demographic data and group of symptom codes with risk factors are integrated. In the preparation phase, the EHRs were later converted to zero or one for defining diseases that the patients suffered (see Table 2).

In the second process, stroke is predicted by using Long Short-Term Memory - Recurrent Neural Network (LSTM-RNN), which is currently the most suitable approach. For prediction algorithm, the dataset was trained by means of feature selection and retrieval process, and LSTM-RNN prediction formula is applied. The input layer calculated the weight values based on ICD-10 codes and EHRs with risk factors of stroke. The ICD-10 codes that represent stroke risk factors are selected by using AHA guideline [16–18]. This group was a knowledge-based reference that was used for computing the weight for embedding at hidden layer as input. The weight values and EHRs were integrated into LSTM-RNN layer. The output layer of prediction model represented the prediction value in a form of percentage risk (see Fig. 2).

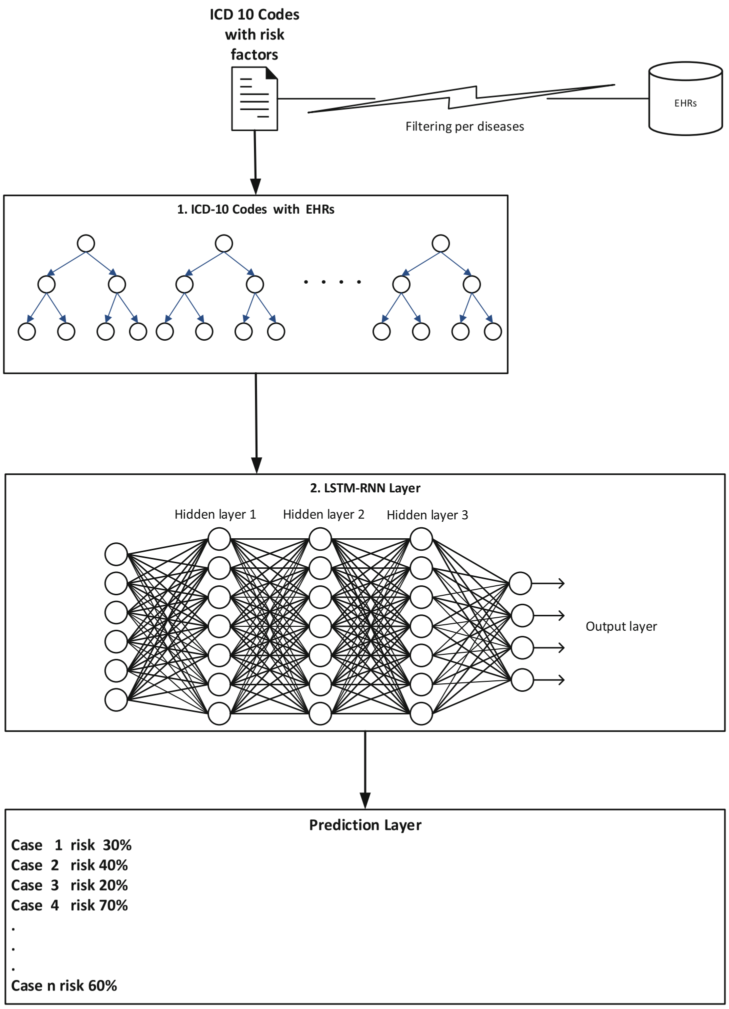


Fig. 2. Model of stroke prediction using EHRs and deep learning

4.1 The Selection of EHRs Based on Risk Factors

The ICD-10 code that had been applied in EHRs can be used for training probabilistic classifiers from the large data sets of EHRs. Specifically, we consider mulitlabel classification of stroke symptoms and risk factors for training and modelling by selection based on AHA list of stroke factors [16–18]. Normally, ICD-10 code presented in main categories and sub-categories such as I65 (Occlusion and stenosis of vertebral artery) is the main category, and I65.1 (Occlusion and stenosis of basilar artery) is the sub-category. The code risk factor that has been chosen consists of 70 main-categories and about 200 sub-categories, all together are 227 factors.

In preparation process after cleaning the data, we reformatted the DOB filed by computed to age of patient. After filtering with ICD-10 codes, these codes will be shown in 1 or 0 to represent the existent of risk factor for each record. The former EHRs with mixed codes then was rearranged to a new EHRs dataset as show in Table 2. This new dataset is smaller in size and suitable for train and test in prediction process.

Table 2. EHRs records with stroke's risk factors.

Age	Gender	B980	E108	E119		I64	Z721	Z920
74	1	1	0	0	……..	0	0	0
65	2	1	0	0	……..	0	0	0
61	1	1	0	1	……..	0	0	0
⋮	⋮	⋮	⋮	⋮		⋮	⋮	⋮
50	1	0	1	1	…….	0	0	0
62	2	0	1	0	…….	0	0	0
76	1	0	1	1	…….	1	0	0
83	2	0	0	1	…….	0	0	0

4.2 Deep Learning by Using Long Short -Term Memory Recurrent Neural Networks (LSTM-RNN)

In this section, we implemented the LSTM-RNN. An architecture contained computation units in each memory block in the recurrent hidden layer. The memory block contained memory cells with self-connections storing the temporal state of the network in addition to multiplicative unit that was called 'gate', which controlled the flow of information inputted to unit. An input gate and output gate were included in the original architecture. The input gate controlled the flow of information and activations into the cell that was computed by *sigmoid* and *tanh* function. The output gate controlled the output flow of cell that activation function was computed by using *sigmoid* and *tanh* function for the rest of the network.

The forget gate was added to the memory block. This gate prevented a weakness of LSTM models from processing continuous input streams that are not segments into subsequences. The internal state of cell of the forget gate scales run verification before adding an input to the cell through the self-recurrent connection of the cell; therefore, it would forget or reset the cell's memory [19]. This gate used *sigmoid* function for computation. Furthermore, in the LSTM architecture peepholes connections (green line) form its internal cells were applied to all gates in the same cell for learning precise timing of the outputs [20] (see Fig. 3).

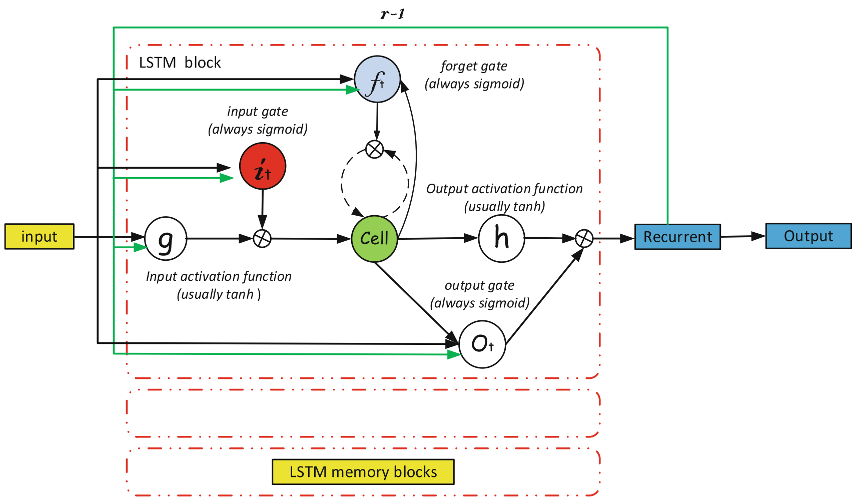

r-1
LSTM block
forget gate (always sigmoid)
input gate (always sigmoid)
Output activation function (usually tanh)
input
g
Cell
h
Recurrent
Output
Input activation function (usually tanh)
output gate (always sigmoid)
LSTM memory blocks

Fig. 3. LSTMP-RNN memory cell architecture and memory blocks [20–22].

5 Evaluation Model

5.1 Data Source

This research used aggregated files of Electronic Healthcare Records (EHRs) from Department of Medical Services, The Ministry of Public Health of Thailand between 2015 and 2016 (326,152 records). It consisted of demographic data, diseases codes (ICD-10 codes), Dates of diagnosis, clinic types, and types of diagnosis (see in Table 1). According to the source, EHRs data had multiple value dependencies that was cleaned anomaly data and filtered by ICD-10 codes for risk factors of stroke. Subsequently, we had a new EHRs dataset (See in Table 2). The new datasets actually had 96,127 records of the stroke patients and non-stroke patients who encountered potential risk factors.

5.2 Predictive Model

The algorithm deep learning relies Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) that is wildly used in prediction. In this research, it was applied to a large scale of an aggregated file from Electronic Healthcare Records. The algorithm model appears as follows:

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh^x t}) \tag{1}$$

$$a_i = \sigma(W_a x_i + W_a h_{i-1} + b_a) \tag{2}$$

$$g_i = \sigma\left(W_g x_i + W_g h_{i-1} + b_g\right) \tag{3}$$

$$\tilde{h}_i = tanh(W_h x_i + g_i \circ W_h h_{i-1} + b_h) \tag{4}$$

$$h_i = a_i \circ h_{i-1} + (1 - g_i) \circ \tilde{h}_i \tag{5}$$

Where the W terms are weight matrices value, W_h, W_b, and W_a, are diagonal weight values for next layer to connections. The b terms are bias vectors. The logistic sigmoid function is represented by σ. The input gate, forget gate, and output gate are represented by a, g, and n respectively. All of them are in the same size as the cell output activation vectors h_i, $\circ$ is the element product of the vector $\tilde{h}_i$ is the cell input and cell output activation function, generally and in this research network is *tanh*.

We initiated the prediction model for training stroke symptom and risk factors based on ICD-10 standard. The equations of the model appear as follows:

$$h_t = \tanh \begin{matrix} (W(I64 \sim Age) + W(I64 \sim Gender) + \\ W(I64 \sim Stroke's\, risk factors)) \end{matrix} \tag{6}$$

The machine learned from model and pattern. The group of codes was computed for finding the weight value in each node. The learning rate term is 0.01 and epoch in 10 and their network type is LSTM.

6 Result

We conducted a comparison between three models: Backpropagation; RNN; and LSTM- RNN. Algorithm backpropagation, RNN, and LSTM-RNN were applied in prediction. All techniques demonstrated the results of training 30%, 50%, and 80% respectively. For testing, 10% of the dataset is used. A learning rate is 0.1 and the number of iteration (10-epochs) were used for prediction. The variable for calculating was used for 227 risk factors for stroke.

The result shows that during training procedure, the accuracy value, precision value, recall value, and F1 scores for prediction in LSTM-RNN are higher than those obtained from the other two techniques. The result of RNN shown for all value lowest at 50% of sample size (0.3570; 0.3612; 0.6476; 0.5456).

In backpropagation, we used a feedforward multilayer artificial neural network. The computation shows that accuracy is at 0.8912, 0.8917, and 0.8914, F1 score as 0.3857, 0.3857, and 0.3860 respectively (shown in Table 3). This method show that all values are not differ in the three sample size for prediction. Only, the accuracy shown here is a little bit changed when the sample data slightly increased.

By using the same parameters as in previous techniques, the LSTM-RNN show the best results the best performance for prediction of stroke. The accuracy is 0.9279, 0.9493, and 0.9998 and F1 score are 0.9626, 0.9738, and 0.9999 respectively. The prediction for stroke will be shown in percentage which mean that change of getting stroke. In medical domain, the good performance is the preferred algorithm and LSTM-RNN is considered confidence to use with huge dataset.

Table 3. Metrics of stroke prediction

Train 30% and Test 10%

Techniques	Precision	Recall	F1	Accuracy
Backpropagation	1	1	0.3857	0.8912
RNN	0.9371	0.9375	0.9371	0.8825
LSTM-RNN	**0.0219**	**0.0216**	**0.9626**	**0.9279**

Train 50% and Test 10%

Techniques	Precision	Recall	F1	Accuracy
Backpropagation	1	1	0.3857	0.8917
RNN	0.3570	0.3612	0.6476	0.5456
LSTM-RNN	**0.9741**	**0.9738**	**0.9738**	**0.9493**

Train 80% and Test 10%

Techniques	Precision	Recall	F1	Accuracy
Backpropagation	1	1	0.3860	0.8914
RNN	0.9996	0.9996	0.9996	0.9992
LSTM-RNN	**0.9999**	**0.9999**	**0.9999**	**0.9998**

7 Conclusion and Future Work

This research aims at the use of deep learning technique and EHRs based on risk factors to predict for cerebrovascular disease by LSTM-RNN algorithm. The Electronic Healthcare Records (EHRs) provide the descriptive details about a patient's physical and mental health, diagnosis, lab results, treatments care plan and so forth. The data are difficult to mine effectively due to irregular sampling and missing data. Nowadays, the diagnoses of disease are represented by International Classification of Diseases, 10th Revision (ICD-10) code in each patient record. It enables researchers to train and develop a model to perform early diagnosis by predicting various risk factors.

The results of using LSTM-RNN show that accuracy rate, recall and F1 measure score are different from those of back propagation and RNN algorithm. Therefore, an accuracy rate depend on the size of sample. Unlike other techniques, the result is more reliable once there are large datasets for the prediction of stroke. This confirms that LSTM algorithm is most suitable for predictive analysis of any cerebrovascular disease or stroke.

EHRs using ICD-10 code have some issues and challenges in the data analyses of various diseases health problems by means of deep learning. The excellent analysis by different predicting techniques require the use of data obtained from patient health records and a comparison between previous cases, observation, or inspection. Stroke has complex risk factors, so algorithms with very high level of accuracy are therefore vital for medical diagnosis. The development of algorithms, nevertheless, still remains obscure despite its importance and necessity for healthcare. Good performance comes along with specific favourable circumstances, for instance, when well designed and formulated inputs are guaranteed. However, the deep learning allows the disclosure of some unknown or unexpressed knowledge during prediction procedure, which is beneficial for decision-making in medical practice and can provide useful suggestions and warnings to patient about unpredictable stroke. In future, we will use more risk factors and lab results to predict using deep learning algorithm and implement to an e-stroke application.

Acknowledgements. This research was granted the ethics approval by University of Technology Sydney, Australia (The ethics approval number UTS HREC ETH17-1406). Additionally, without support from the Department of Medical Services (affiliated with Ministry of Public Health of Thailand) and Office of Educational Affairs (affiliated with Royal Thai Embassy to Australia), the research would not have been accomplished. With this acknowledgement, we would like to express our sincere appreciation to them.

References

1. Deo, R.C.: Machine learning in medicine. Circulation **132**, 1920 (2015)
2. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**, 436–444 (2015)
3. Langhorne, P., Bernhardt, J., Kwakkel, G.: Stroke rehabilitation. Lancet **377**, 1693–1702 (2011)
4. Khosla, A., Cao, Y., Lin, C.C.-Y., Chiu, H.-K., Hu, J., Lee, H.: An integrated machine learning approach to stroke prediction. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 183–192. ACM, Washington, D.C. (2010)
5. Leira, E.C., Ku-Chou, C., Davis, P.H., Clarke, W.R., Woolson, R.F., Hansen, M.D., Adams Jr., H.P.: Can we predict early recurrence in acute stroke? Cerebrovasc. Dis. **18**, 139–144 (2004)
6. Cooke, C.R., Joo, M.J., Anderson, S.M., Lee, T.A., Udris, E.M., Johnson, E., Au, D.H.: The validity of using ICD-9 codes and pharmacy records to identify patients with chronic obstructive pulmonary disease. BMC Health Serv. Res. **11**, 37 (2011)
7. Ried, L.D.P., Cameon, R.M.S., Jia, H.P., Findley, K., Hinojosa, M.S.P., Wang, X.P., Tueth, M.J.M.D.: Identifying veterans with acute strokes with high-specificity ICD-9 algorithm with VA automated records and Medicare claims data: a more complete picture. J. Rehabil. Res. Dev. **44**, 665–673 (2007)
8. Scheurer, D.B., Hicks, L.S., Cook, E.F., Schnipper, J.L.: Accuracy of ICD-9 coding for Clostridium difficile infections: a retrospective cohort. Epidemiol. Infect. **135**, 1010–1013 (2007)
9. WHO: ICD10: International Statistical Classification of Disease and Related Health Tenth Revision, vol. 2. World Health Organization, Geneva (2004)

10. Gulshan, V., Peng, L., Coram, M., et al.: Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. JAMA **316**, 2402–2410 (2016)
11. Stier, N., Vincent, N., Liebeskind, D., Scalzo, F.: Deep learning of tissue fate features in acute ischemic stroke. In: IEEE International Conference on Bioinformatics and Biomedicine (BIBM) 2015, pp. 1316–1321 (2015)
12. Liang, Z., Zhang, G., Huang, J.X., Hu, Q.V.: Deep learning for healthcare decision making with EMRs. In: IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 556–559 (2014)
13. Hammerla, N.Y., Fisher, J., Andras, P., Rochester, L., Walker, R., Plötz, T.: PD disease state assessment in naturalistic environments using deep learning. In: Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 1742–1748 (2015)
14. Ding, X., Zhang, Y., Liu, T., Duan, J.: Deep learning for event-driven stock prediction. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015), pp. 2327–2333 (2015)
15. Ding, X., Zhang, Y., Liu, T., Duan, J.: Using structured events to predict stock price movement: an empirical investigation. In: EMNLP, pp. 1415–1425 (2014)
16. Goldstein, L.B., Adams, R., Becker, K., Furberg, C.D., Gorelick, P.B., Hademenos, G., Hill, M., Howard, G., Howard, V.J., Jacobs, B., Levine, S.R., Mosca, L., Sacco, R.L., Sherman, D.G., Wolf, P.A., del Zoppo, G.J.: Members: primary prevention of ischemic stroke. Circulation **103**, 163–182 (2001)
17. Goldstein, L.B., Adams, R., Alberts, M.J., Appel, L.J., Brass, L.M., Bushnell, C.D., Culebras, A., DeGraba, T.J., Gorelick, P.B., Guyton, J.R., Hart, R.G., Howard, G., Kelly-Hayes, M., Nixon, J.V., Sacco, R.L.: Primary prevention of ischemic stroke: a guideline from the American Heart Association/American Stroke Association Stroke Council: cosponsored by the Atherosclerotic Peripheral Vascular Disease Interdisciplinary Working Group; Cardiovascular Nursing Council; Clinical Cardiology Council; Nutrition, Physical Activity, and Metabolism Council; and the Quality of Care and Outcomes Research Interdisciplinary Working Group: the American Academy of Neurology affirms the value of this guideline. Stroke **37**, 1583–1633 (2006)
18. The American Heart Association: Comparison of 12 risk stratification schemes to predict stroke in patients with nonvalvular atrial fibrillation. Stroke **39**, 1901–1910 (2008)
19. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with LSTM. Neural Comput. **12**, 2451–2471 (2000)
20. Gers, F.A., Schraudolph, N.N., Schmidhuber, J.: Learning precise timing with LSTM recurrent networks. J. Mach. Learn. Res. **3**, 115–143 (2002)
21. Sak, H., Senior, A., Beaufays, F.: Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In: Fifteenth Annual Conference of the International Speech Communication Association, Singapore, pp. 338–342 (2014)
22. Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J.: LSTM: a search space odyssey. IEEE Trans. Neural Netw. Learn. Syst. **28**, 2222–2232 (2017)

Adversarial Machine Learning: A Literature Review

Sam Thomas(✉) and Nasseh Tabrizi

East Carolina University, Greenville, NC 27858, USA
thomass08@students.ecu.edu, tabrizim@ecu.edu

Abstract. Machine learning is becoming more and more utilized as a tool for businesses and governments to aid in decision making and automation processes. These systems are also susceptible to attacks by an adversary, who may try evading or corrupting the system. In this paper, we survey the current landscape of research in this field, and provide analysis of the overall results and of the trends in research. We also identify several topics which can better define the categorization.

Keywords: Adversarial machine learning · Literature survey

1 Introduction

Machine learning is the process by which a machine can learn to make decisions without being explicitly told what to do. This has been a boon to automation and data science as a whole. However, machine learning systems are not inherently robust. An entity that wishes to harm or evade an unguarded system's decision-making capabilities can do so with relative ease. This is the conceptual foundation of adversarial machine learning [1].

Adversarial machine learning (AdvML), broadly speaking, is where a machine learning system (i.e. the classifier) is in an adversarial environment – one in which it is challenged by some adversarial opponent. These opponent input samples, which have been designed to disrupt the performance of the ML system, alter the legitimate input samples by tricking the classifier into misclassifying the input. In fact, according to [1] machine learning systems can, and often are, trained to generate adversarial samples to use against the classifier Although the measures can be taken to protect a machine learning system, the protection is not total and not ensured to last. Thus, as reported in [2] this still remains an open problem.

Generally, AdvML is applicable wherever there is a machine learning system that is accessible to would-be attackers. Notably, it has applications in biometric verification, spam detection, malware detection, and the detection of network intrusions. In addition, generative adversarial networks (GAN) [1, 4, 5] have shown that they can be applied to a wide variety of tasks, such as medical image processing, image censoring, language generation, and learning representations of emotional speech, to name a few. These applications go beyond the scope of simply protecting a classifier from adversarial examples.

As stated earlier, if a classifier is not protected from adversaries, then it is very susceptible to their attacks, resulting in misclassification rate of over 96% [3, 4, 8]. On-line machine learning systems are susceptible to being corrupted over time, and have been shown to become more inaccurate as the number of adversarial samples increases [6], and as reported in [7] there is also the ability to fool autonomous robotic patrolling by using game-theoretic approaches.

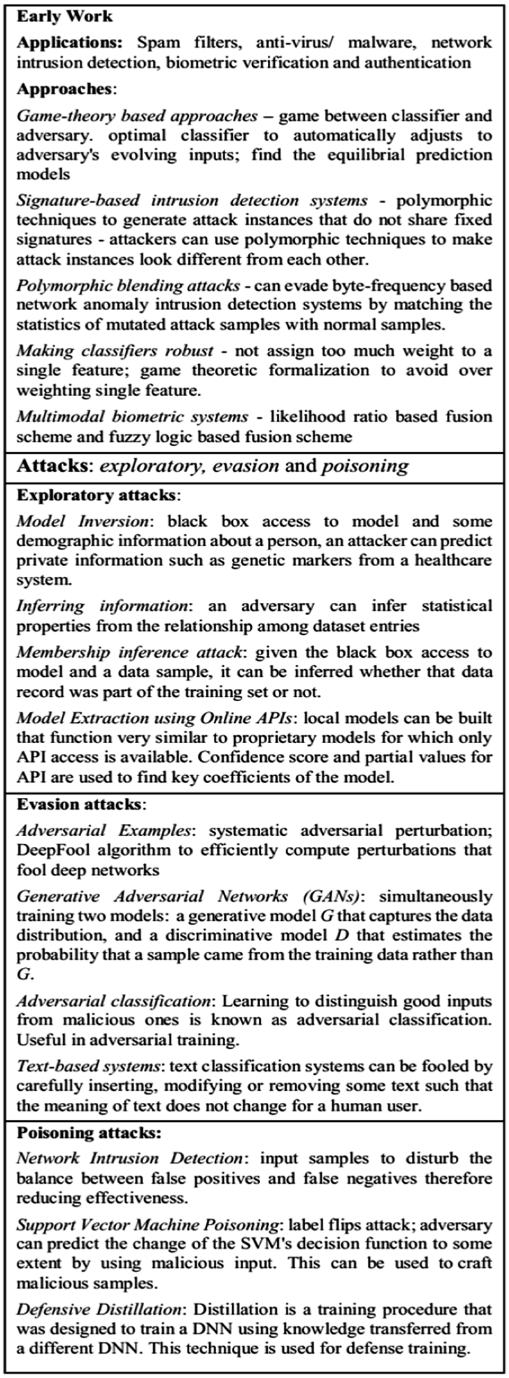
Our research offers the following contributions:

- A survey of research papers on the subject of "adversarial machine learning". The papers are counted and sorted by categories, and that data is compared and trends are analyzed.
- A refinement of the categorization of topics within the field of adversarial machine learning.

2 Related Works

This paper adapts the categorization used in a survey done by Kumar and Mehta of IBM Research, India [9]. The categories and descriptions can be seen in Fig. 1. We have outline the categories below, which include the adaptation we made for the purposes of this literature review. This categorization was chosen because it provides the basis for a deeper taxonomy than has been proposed by any other paper we found.

The categories and subcategories that we used are as follows:



Early Work

Applications: Spam filters, anti-virus/ malware, network intrusion detection, biometric verification and authentication

Approaches:

Game-theory based approaches – game between classifier and adversary. optimal classifier to automatically adjusts to adversary's evolving inputs; find the equilibrial prediction models

Signature-based intrusion detection systems - polymorphic techniques to generate attack instances that do not share fixed signatures - attackers can use polymorphic techniques to make attack instances look different from each other.

Polymorphic blending attacks - can evade byte-frequency based network anomaly intrusion detection systems by matching the statistics of mutated attack samples with normal samples.

Making classifiers robust - not assign too much weight to a single feature; game theoretic formalization to avoid over weighting single feature.

Multimodal biometric systems - likelihood ratio based fusion scheme and fuzzy logic based fusion scheme

Attacks: *exploratory, evasion* and *poisoning*

Exploratory attacks:

Model Inversion: black box access to model and some demographic information about a person, an attacker can predict private information such as genetic markers from a healthcare system.

Inferring information: an adversary can infer statistical properties from the relationship among dataset entries

Membership inference attack: given the black box access to model and a data sample, it can be inferred whether that data record was part of the training set or not.

Model Extraction using Online APIs: local models can be built that function very similar to proprietary models for which only API access is available. Confidence score and partial values for API are used to find key coefficients of the model.

Evasion attacks:

Adversarial Examples: systematic adversarial perturbation; DeepFool algorithm to efficiently compute perturbations that fool deep networks

Generative Adversarial Networks (GANs): simultaneously training two models: a generative model *G* that captures the data distribution, and a discriminative model *D* that estimates the probability that a sample came from the training data rather than *G*.

Adversarial classification: Learning to distinguish good inputs from malicious ones is known as adversarial classification. Useful in adversarial training.

Text-based systems: text classification systems can be fooled by carefully inserting, modifying or removing some text such that the meaning of text does not change for a human user.

Poisoning attacks:

Network Intrusion Detection: input samples to disturb the balance between false positives and false negatives therefore reducing effectiveness.

Support Vector Machine Poisoning: label flips attack; adversary can predict the change of the SVM's decision function to some extent by using malicious input. This can be used to craft malicious samples.

Defensive Distillation: Distillation is a training procedure that was designed to train a DNN using knowledge transferred from a different DNN. This technique is used for defense training.

Fig. 1. Category summary, adapted from [9].

- Applications
  - Spam Filters
  - Anti-virus/Malware Detection
  - Network Intrusion Detection
  - Biometric Verification and Authentication
  - Ill-Fit
- Approaches
  - Game-theoretic Approaches
  - Signature-based Intrusion Detection Systems
  - Polymorphic Blending Attacks
  - Making Classifiers Robust
  - Multimodal Biometric Systems
  - Ill-Fit
- Attacks
  - Exploratory Attacks:
    - Model Inversion
    - Inferring Information
    - Membership Inference Attack
    - Model Extraction using Online APIs
    - Ill-Fit
  - Evasion Attacks:
    - Adversarial Examples
    - Generative Adversarial Networks (GANs)
    - Adversarial Classification
    - Text-based Systems
    - Ill-Fit
  - Poisoning Attacks:
    - Network Intrusion Detection
    - Support Vector Machine Poisoning
    - Defensive Distillation
    - Ill-Fit
  - Ill-Fit

3 Methodology

3.1 Overview

Our process for categorizing recent research papers is as follows:

1. Collect the top 100 results from the sources, using the phrase ‘adversarial machine learning’ as the search query.
2. Review each paper.
3. Sort each paper into categories.
4. Tally the raw values of each category.

5. Tally the per-year values of each category (excluding Cornell Digital Library [https://arxiv.org/]).
6. Visualize the data with charts.

3.2 Collection

In this study we have used multiple collection sources to get a robust data set. The sources we used were ACM Digital Library, IEEE Xplore Digital Library, Springerlink, Cornell Digital Library, and Sciencedirect. Since we are focused on newer developments within this field of research, we restricted our data collection to the past ten years (2007–2017).

3.3 Data-Set Restrictions

Here we have listed the details of our collection methodology as a whole. Also, in order to get accurate results, some sources needed search criteria that was specific to them, and so those criteria are listed here, as well as any quirks specific to those sources.

General

- Collection took place: 8/28/17–9/11/17
- Language: English
- Papers published: 2007–2017
- Our university's access to the collections was used to gather the papers.
- Only those papers which were acquirable were collected (i.e. no "abstract only" or pay-to-access entries)
- Papers that are not about adversarial machine learning, but merely mention it, have been classified as "unrelated". These papers were collected, but during evaluation they have been sorted under "discarded".

IEEE Xplore Digital Library

- Only 91 papers matched the search query

Cornell Digital Library

- Experimental full text search in the subject Computer Science

Springer Link

- Searched under the discipline of Computer Science
- Did not include "Preview-Only Content"
- Sorted by relevance

3.4 Evaluation

All papers were filtered and categorized manually. We filtered out book chapters, conference posters, and those papers which were unrelated to the subject matter. We used the categories in such a way, that a paper can appear in multiple categories. Some papers were categorized as a top-level category, but did not fit into any of its subcategories. We tallied these papers in their own subcategory that is independent of the other subcategories.

The histograms for each category exclude the results from Cornell Digital Library due to the fact that the earliest published papers are from 2015, and so including these results would heavily skew the graph.

4 Results

Here we present the data that we have collected. These have been divided into sections for "Pre-Sort", "Raw Counts", and "Trends".

4.1 Pre-sort

The first step was to refine the data set which would be sorted. For this purpose, we used the following categories:

- Disregard: disregarded papers, such as conference posters, conference abstracts, and book chapters.
- Duplicates: duplicate papers.
- Meta: papers such as surveys, literature reviews, or topic overviews.
- Tools: papers which only discussed a tool that was being demonstrated.
- Unrelated: papers that are unrelated to "Adversarial Machine Learning".
- Sorted: those papers which would be categorized.

The results of these refinements can be seen in Fig. 2.

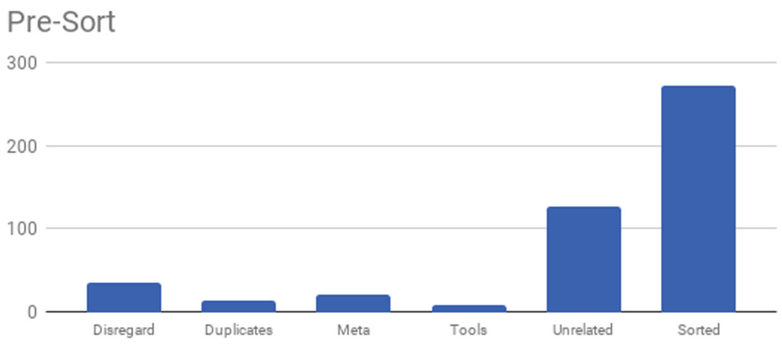


Fig. 2. Count of papers for the Pre-Sort step.

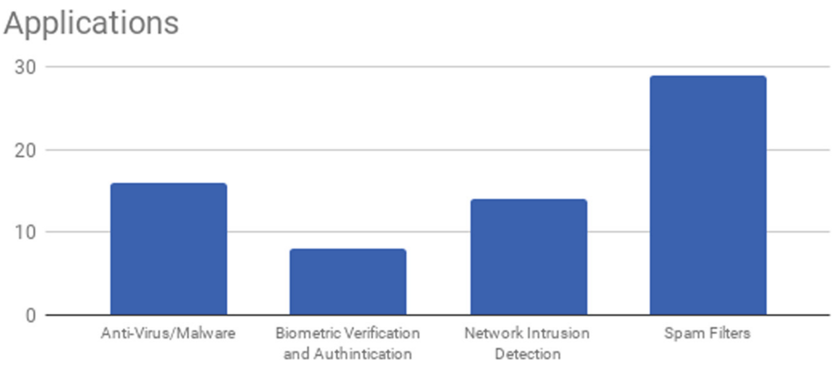


Fig. 3. Count of papers for "Applications"

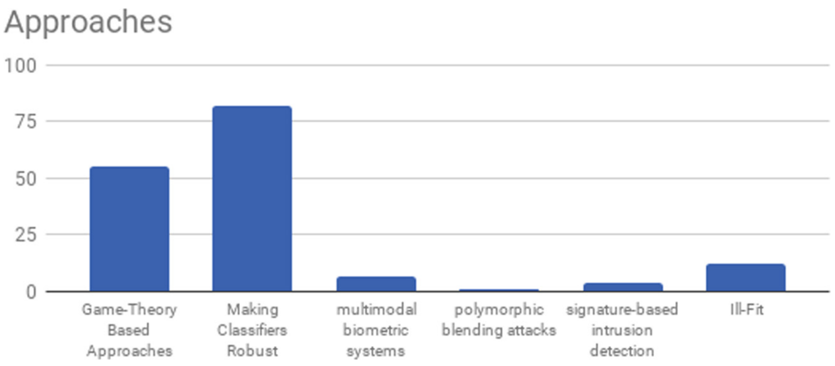


Fig. 4. Count of papers for "Approaches"

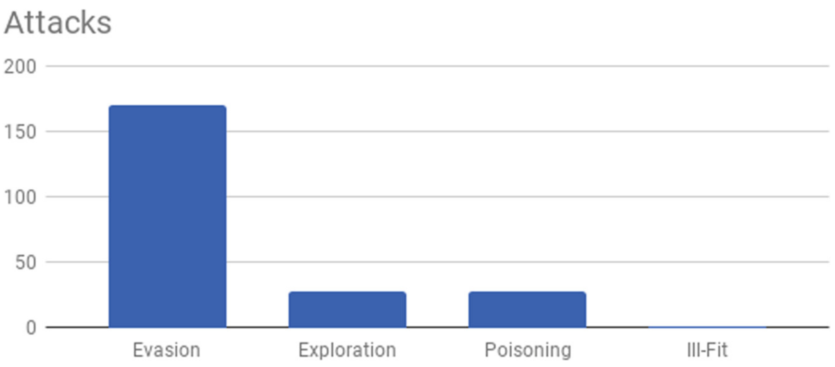


Fig. 5. Count of papers for "Attacks"

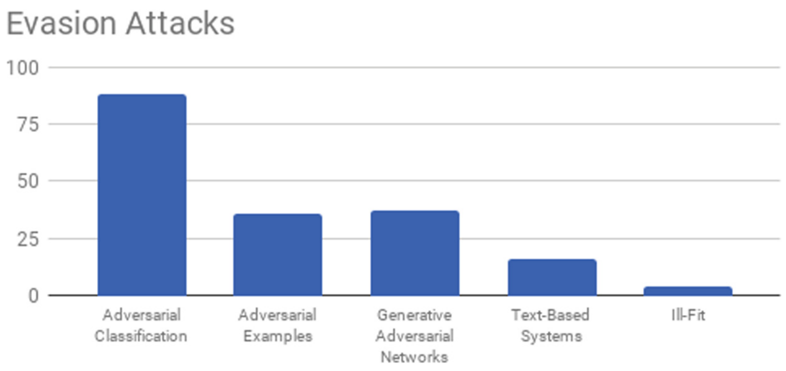


Fig. 6. Count of papers for "Evasion Attacks" sub-category

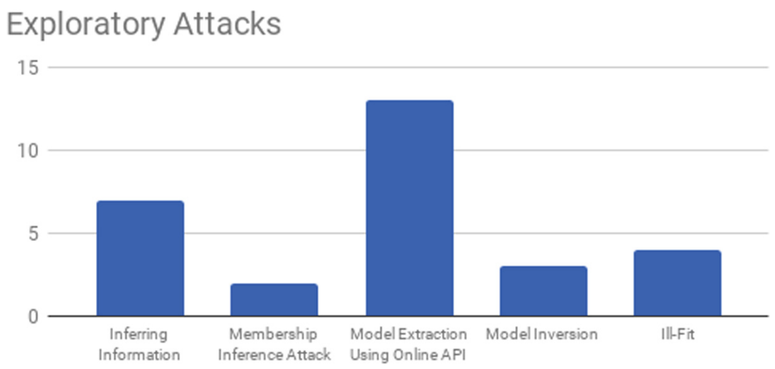


Fig. 7. Count of papers for "Exploratory Attacks" sub-category

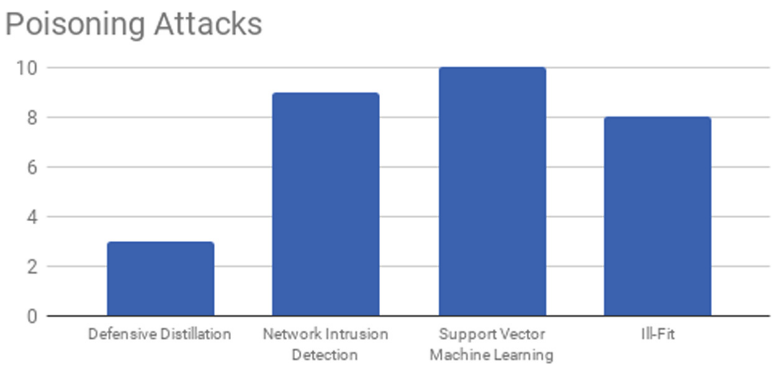


Fig. 8. Count of papers for "Poisoning Attacks" sub-category

4.2 Raw Counts

The raw paper counts for the subcategories are shown in Figs. 3, 4, 5, 6, 7 and 8. Figures 3, 4 and 5 show the counts for top-level categories, while Figs. 6, 7 and 8 show the counts for each type of attack.

4.3 Trends

The counts per year can be seen in Figs. 9, 10, 11, 12, 13, 14, 15 and 16. These figures are a subset of all the trends which were analyzed. For the sake of readability and brevity, we have included the most noteworthy trend charts in this paper, and excluded those which had sparse or sporadic paper counts.

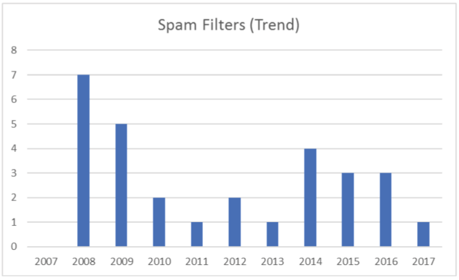


Fig. 9. Paper counts by year for "Spam Filters".

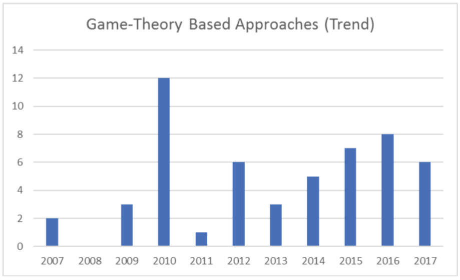


Fig. 10. Paper counts by year for "Game-Theory Approaches".

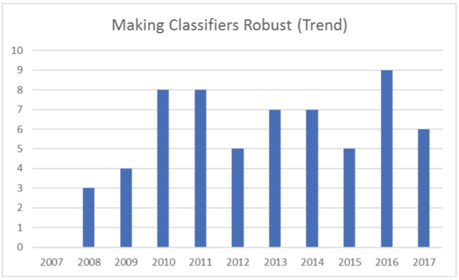


Fig. 11. Paper counts by year for "Making Classifiers Robust".

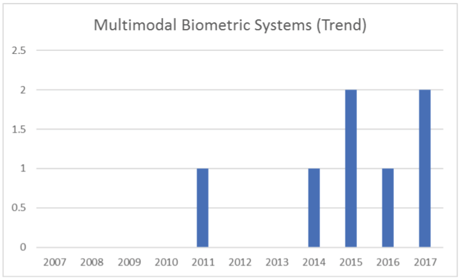


Fig. 12. Paper counts by year for "Multimodal Biometric Systems".

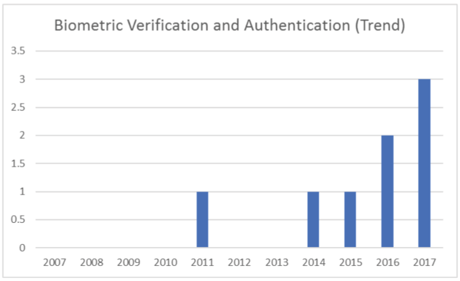


Fig. 13. Paper counts by year for "Biometric Verification and Authentication".

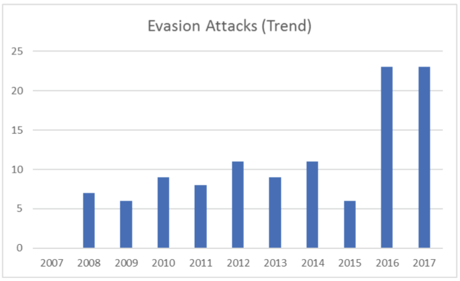


Fig. 14. Paper counts by year for "Evasion Attacks".

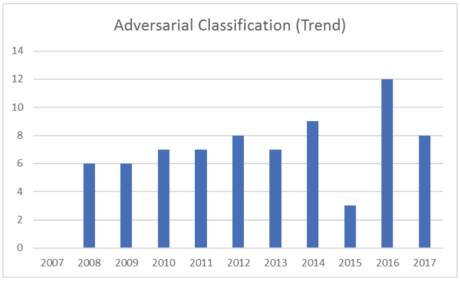


Fig. 15. Paper counts by year for "Adversarial Classification".

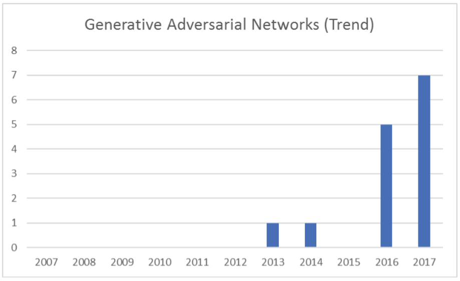


Fig. 16. Paper counts by year for "Generative Adversarial Networks".

5 Discussion

5.1 Raw Count Data Analysis

From Figs. 3, 4, 5, 6, 7 and 8 we can see the following:

- "Spam Detection" is by far the most popular application, with ~41% of papers exploring the topic.
- "Making Classifiers Robust" and "Game-Theory Based Approaches" are the most common approaches, making up approximately 51% and 34% of the papers, respectively.
- ~76% of papers discussing attacks focus on "Evasion" attacks.
- Regarding evasion attacks, the most explored topic was "Adversarial Classification", have ~49% of papers which discuss it. There were also the topics of "Adversarial Examples" and "Generative Adversarial Networks" which were somewhat popular, at ~20%, each.
- Regarding exploratory attacks, the most explored aspect was "Model Extraction Using Online APIs", which made up ~45% of papers in that category.

- For the "Ill-Fit" category, we see that "Poisoning Attacks" has the largest proportion of ill-fit papers, with $\sim 27\%$ falling into the sub-category. However, it is the "Approaches" category which has the highest raw count, at 12 ill-fitting papers.

These are some of the notable topics that were discovered during evaluation but were designated as "ill-fit":

- *Applications*: Privacy
- *Approaches*: Feature Squeezing
- *Approaches*: Domain Adaptation
- *Poisoning Attacks*: Online Neural Networks.

5.2 Trends Data Analysis

By looking at the breakdown of each category's paper-counts by year, we can see the following trends in Figs. 9, 10, 11, 12, 13, 14, 15 and 16:

- "Spam Filters" had a large spike in 2008, before declining, and then rising slightly in popularity.
- "Game-Theory Based Approaches" had a large spike in 2010, before lowering and remaining stable.
- "Making Classifiers Robust" has been a consistently explored topic since 2008.
- Figures 12, 13 show that AdvML in biometric systems has become more popular in recent years.
- Evasion attacks have had steady interest since 2008, with a sharp rise within the past 2 years.
- "Adversarial Classification" has had steady interest since 2008.
- "Generative Adversarial Networks" have garnered a large amount of interest in just the past few years.

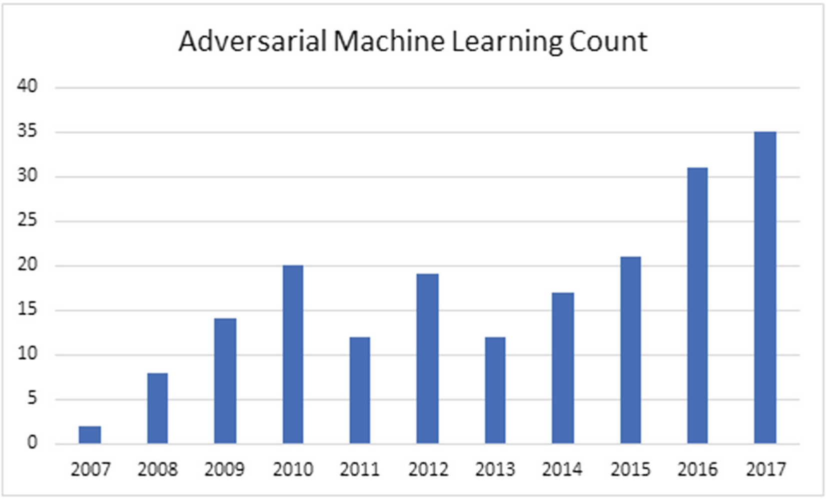


Fig. 17. Paper counts by year for all papers in the field of Adversarial Machine Learning

And finally, we see in Fig. 17 the 10-year trend in the topic of Adversarial Machine Learning as a whole.

Together, these charts show an initial wave of interest, which peaks in 2010. Interest in the field levels off for a few years, but then begins to steadily gain more and more interest from 2013 onward.

6 Conclusion

In this paper, we have presented data on the current state and trends of the field of "Adversarial Machine Learning". To this end, we collected, sorted, and analyzed 475 research papers from five different sources.

We found that certain topics are more popular than others, and that some topics have only recently started to gain interest within this field.

Notably, we have identified that, while there has been steady interest in the subject for a while, AdvML is rapidly gaining in popularity, and that this popularity is evident in the trends of lesser-researched topics within the field.

We also identified four topics within the field which should be regarded when categorizing the subject. These topics are privacy, feature squeezing, domain adaptation, and online neural networks.

References

1. Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I.P., Tygar, J.D.: Adversarial machine learning. In: Proceedings of the 4th ACM Workshop on Security Artificial Intelligence, pp. 43–58 (2011)
2. Papernot, N., McDaniel, P., Wu, X., Jha, S., Swami, A.: Distillation as a defense to adversarial perturbations against deep neural networks. In: Proceedings - 2016 IEEE Symposium on Security and Privacy, SP 2016, pp. 582–597 (2016)
3. Papernot, N., McDaniel, P., Goodfellow, I.: Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. CoRR abs/1605.07277 (2016)
4. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning. CoRR abs/1602.02697(2016)
5. Biggio, B., Fumera, G., Roli, F.: Adversarial pattern classification using multiple classifiers and randomisation. In: da Vitoria Lobo, N., et al. (eds.) SSPR /SPR 2008. LNCS, vol. 5342, pp. 500–509. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89689-0_54
6. Biggio, B., Corona, I., Fumera, G., Giacinto, G., Roli, F.: Bagging classifiers for fighting poisoning attacks in adversarial classification tasks. In: Sansone, C., Kittler, J., Roli, F. (eds.) MCS 2011. LNCS, vol. 6713, pp. 350–359. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21557-5_37
7. Villacorta, P.J., Pelta, D.A.: Exploiting adversarial uncertainty in robotic patrolling: a simulation-based analysis. In: Greco, S., et al. (eds.) IPMU 2012 Part IV. CCIS, vol. 300, pp. 529–538. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31724-8_55
8. Papernot, N., Mcdaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: Proceedings - 2016 IEEE European Symposium Security Privacy, EURO S P 2016, pp. 372–387 (2016)
9. Kumar, A., Mehta, S.: A survey on resilient machine learning. CoRR, abs/1707.03184 (2017)

Reverse Engineering Gene Regulatory Networks Using Graph Mining

Haodi Jiang[1], Turki Turki[2(✉)], Sen Zhang[3], and Jason T. L. Wang[1(✉)]

[1] Bioinformatics Program and Department of Computer Science, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA
wangj@njit.edu

[2] Computer Science Department, King Abdulaziz University, P.O. Box 80221, Jeddah 21589, Saudi Arabia
tturki@kau.edu.sa

[3] Department of Mathematics, Computer Science and Statistics, State University of New York (SUNY) College at Oneonta, Oneonta, NY 13820, USA

Abstract. Reverse engineering gene regulatory networks (GRNs), also known as GRN inference, refers to the process of reconstructing GRNs from gene expression data. A GRN is modeled as a directed graph in which nodes represent genes and edges show regulatory relationships between the genes. By predicting the edges to infer a GRN, biologists can gain a better understanding of regulatory circuits and functional elements in cells. Many bioinformatics tools have been developed to computationally reverse engineer GRNs. However, none of these tools is able to perform perfect GRN inference. In this paper, we propose a graph mining approach capable of discovering frequent patterns from the GRNs inferred by existing methods. These frequent or common patterns are more likely to occur in true regulatory networks. Experimental results on different datasets demonstrate the good quality of the discovered patterns, and the superiority of our approach over the existing GRN inference methods.

Keywords: Graph mining · Network inference · Pattern discovery
Applications in biology and medicine

1 Introduction

Current biotechnology has allowed researchers in various fields to obtain immense amounts of experimental information, ranging from macromolecular sequences, gene expression data to proteomics and metabolomics. In addition to large-scale genomic information obtained through such methods as third generation DNA sequencing, newer technologies, such as RNA-seq and ChIP-seq, have allowed researchers to fine tune the analysis of gene expression patterns [3]. More information on interactions between transcription factors and DNA, both qualitative and quantitative, is increasingly emerging from the genomics data.

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 335–349, 2018.
https://doi.org/10.1007/978-3-319-96136-1_27

Often, the attachment of transcription factors (TFs) and their binding sites (TFBSs), located at specific gene promoters, influences transcription and modulates RNA production from a particular gene, thus establishing a first level of functional interaction. Since the TFs are gene-encoded polypeptides and the target TFBSs belong to different genes, analyses of TFs-TFBSs interactions could uncover gene networks and may even contribute to elucidate unknown GRNs [20]. Besides contributing to infer and understand these interactions, determining GRNs also aims to provide explanatory models of such connections [4]. GRNs could be the basis to infer more complex networks, encompassing gene, protein, and metabolic spaces, as well as the entangled and often overlooked signaling pathways that interconnect them [5,9,15].

Many bioinformatics tools have been developed to reconstruct GRNs from gene expression data. However, none of these tools is able to perform perfect GRN inference. The accuracy of the predicted GRNs is usually low. To enhance the accuracy, we propose a graph mining approach, which aims to find frequent patterns in the GRNs inferred by existing methods. These frequent or common patterns are more likely to occur in true regulatory networks.

Specifically, we model a GRN by a directed graph G [1,17]. In G, a node represents a gene or transcription factor (TF). An edge from a TF to a gene indicates that the TF regulates the expression of the gene. We consider two types of patterns in G. The first type is an induced subgraph of G, referred to as an induced pattern of G. The second type is a general subgraph of G, referred to as a general pattern of G. Figure 1(a) illustrates a directed graph G, which represents a partial GRN of yeast. G contains five nodes and five edges, where each node represents a yeast gene. Figure 1(b) shows an induced pattern of the graph G in Fig. 1(a). The induced pattern in Fig. 1(b) has three nodes, and consists of all the edges of G with both end nodes in the induced pattern. Notice that, if we remove the edge from the node or gene YMR043W to the node or gene YLR131C, we would obtain a general subgraph of G that is not induced, as shown in Fig. 1(c). Thus, the subgraph in Fig. 1(c) is a general, but not an induced, pattern of G.

Graph mining aims to find frequent or common patterns in multiple graphs. Here we propose new graph mining algorithms to find frequent induced or general patterns in multiple GRNs. Our algorithms are extensions of the Apriori algorithm for frequent item set mining [2,22]. Specifically, we take multiple GRNs produced by three widely-used GRN inference methods including TD-ARACNE [23], GENIE3 [7] and Jump3 [8]. Then, we use the proposed graph mining algorithms to find frequent patterns, induced or general, in these GRNs. Edges in the frequent patterns constitute the GRNs inferred by our approach. Finally we show experimentally that our approach yields more accurate GRNs than those constructed by the three existing GRN inference methods, demonstrating the good quality of the patterns found by our graph mining algorithms.

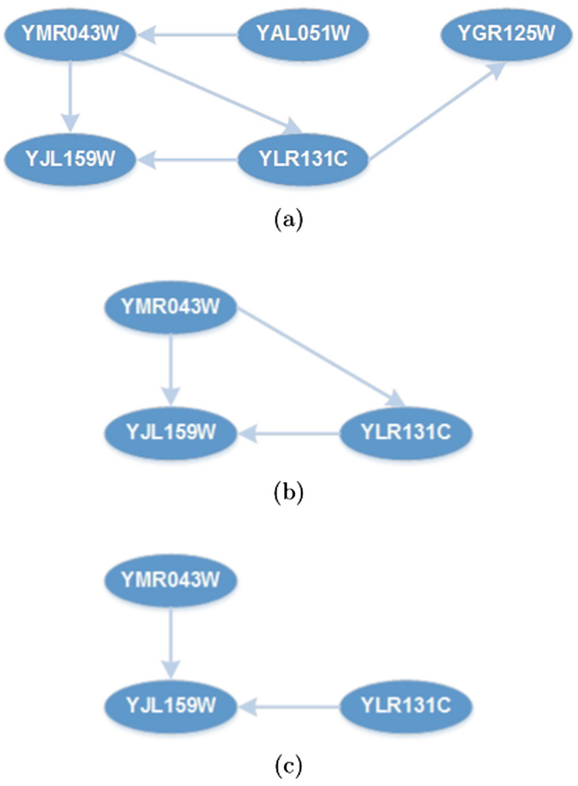


Fig. 1. (a) A directed graph G representing a partial GRN of yeast. (b) An induced pattern of G. (c) A general pattern of G.

1.1 Related Work

A variety of graph mining algorithms have been developed since the early 2000s. AGM (Apriori-based Graph Mining) [10] is the first frequent subgraph mining algorithm that uses a pairwise-join based pattern growth strategy to generate frequent patterns. The algorithm employs the Apriori principle [2] and generates candidate subgraphs of size $(k+1)$, i.e., $(k+1)$-subgraphs, by joining two frequent k-subgraphs that share the same $(k-1)$-subgraphs. FSG [13] is another frequent subgraph mining algorithm, which is similar to AGM in that both algorithms grow patterns level by level through a pairwise join method. However, FSG grows patterns by edges while AGM grows patterns by vertices. FSG finds all frequent connected subgraphs in multiple undirected graphs.

The gSpan [21] program is arguably the most widely used tool for graph mining. The gSpan algorithm grows undirected connected subgraphs in a depth first search (DFS) manner by adding an edge to each possible position on the rightmost path of a known frequent subgraph. Specifically, the algorithm uses DFS lexicographic ordering to construct a tree-like lattice over all possible patterns, resulting in a hierarchical search space called a DFS code tree. Each node of this search tree represents a DFS code. This search tree is traversed in a DFS manner and all subgraphs with non-minimal DFS codes are pruned so that redundant candidate generations are avoided. With the assistance of the DFS code tree, gSpan can discover connected patterns from undirected graphs efficiently.

FFSM [6] focuses on graphs that are dense with a small number of labels. FFSM adopts the same canonical adjacency matrix (CAM) representation used by AGM. A tree-like structure, namely a suboptimal CAM tree, is constructed to include all possible patterns. Each node in this suboptimal CAM tree is created by either a join or an extension operation. FFSM maintains embedding lists for the discovered patterns to avoid explicit subgraph isomorphism testing in the support counting phase [11].

Gaston [16] is a unified graph, sequence and tree extraction algorithm. Given a database of graphs, Gaston finds all frequent subgraphs by using a level-wise approach in which simple paths are first considered, then more complex trees and finally the most complex graphs are considered. Consequently the subgraph mining procedure is invoked only when needed. To determine the frequency of a graph, Gaston employs an occurrence list based approach in which all occurrences of a small set of graphs are stored in main memory. Gaston also maintains embedding lists when growing patterns, to avoid unnecessary subgraph isomorphism testing.

The limitations of these existing graph mining methods are that they are mainly designed and implemented for finding frequent general, not induced, subgraphs in undirected graphs. Thus, the existing graph mining methods are not applicable to GRNs, which are directed graphs. By contrast, our work focuses on finding frequent patterns, induced or general, in directed graphs, more precisely GRNs. Consequently, our pattern growth procedure is different from those employed by the existing graph mining methods.

The rest of this paper is organized as follows. Section 2 describes our Apriori-like pattern finding algorithms. Section 3 reports experimental results, comparing our approach of using graph mining for GRN inference with the three existing GRN inference methods, namely TD-ARACNE [23], GENIE3 [7] and Jump3 [8]. Section 4 concludes the paper and points out some directions for future research.

2 Methods

2.1 Definitions and Notation

We use the terms "graphs" ("subgraphs", "nodes" respectively) and "GRNs" ("patterns", "genes" respectively) interchangeably throughout the paper. All graphs considered here are directed (i.e., the direction of an edge is important). In a species, there are a finite number of genes among which there is no duplicate gene. Thus, each graph has a finite number of nodes and edges, and each node is uniquely labeled.

We do not consider auto-regulation in the paper, though our work can be easily generalized to handle the auto-regulation case. Thus, graphs here have no self-edges (i.e., edges that connect nodes to themselves), neither do the graphs have duplicate edges between two nodes (i.e., two or more edges that connect the two nodes in the same direction). Below we present some definitions and terms used by our algorithms, called *induced pattern discovery* (IPD) and *general pattern discovery* (GPD) algorithms respectively.

Definition 1 (Gene Regulatory Network). A *Gene Regulatory Network* or *GRN* for short is a directed graph $G = (V, E)$ where (i) $V = \{g_1, g_2, \ldots, g_n\}$ is a set of genes, and (ii) $E \subset V \times V$ is a set of edges in which $e_{ij} = (g_i, g_j)$ denotes the edge from gene g_i to gene g_j.

The number of genes in G, i.e., the cardinality of V, denoted by $|V| = n$, is called the *size* of G. A k-pattern is a pattern of size k (i.e., the pattern contains k genes).

Definition 2 (Induced Pattern and General Pattern). Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ where $|V_1| \leq |V_2|, G_1$ is said to be an *induced pattern* (*general pattern*, respectively) of G_2 if there is an injective function f that maps the genes of G_1 to the genes of G_2 such that (i) for each gene g_i in V_1, there is a gene $f(g_i)$ in V_2, and (ii) for any ordered pair of genes g_i, g_j in V_1, edge (g_i, g_j) is in E_1 if and only if (only if, respectively) edge $(f(g_i), f(g_j))$ is in E_2. All patterns considered here are weakly connected, i.e., there is an undirected path between any pair of genes in a pattern.

Definition 3 (Support of a Pattern). Given a set $\mathcal{D}$ of graphs, the *support* of an induced (general, respectively) pattern P, denoted $sup(P)$, is the number of graphs $G \in \mathcal{D}$ where P is an induced (general, respectively) pattern of G.

Definition 4 (Frequent Pattern Discovery). Given a set $\mathcal{D}$ of graphs and a user-specified minimum support threshold (*minsup*), a pattern P in $\mathcal{D}$ is *frequent* if the support of P is greater than or equal to *minsup*. The *frequent induced (general, respectively) pattern discovery* problem is to find all frequent induced (general, respectively) patterns in $\mathcal{D}$.

Our IPD (GPD, respectively) algorithm aims to solve the frequent induced (general, respectively) pattern discovery problem. Below we introduce some data structures needed by our algorithms.

Definition 5 (Gene-Adjacency Matrix and Gene Array). Given a graph $G = (V, E)$ with n genes $g_1, g_2, \ldots, g_n$, the *gene-adjacency matrix* of G, denoted $A(G)$, is a square $n \times n$ matrix whose rows and columns correspond to the edges in G such that the element at the ith row and the jth column of $A(G)$ is one if there is an edge from gene g_i to gene g_j, and is zero otherwise. The array holding the genes, denoted $[g_1, g_2, \ldots, g_n]$, is called the *gene array* of G.

Notice that, since the graphs we consider here do not have self-edges, the diagonal entries of all gene-adjacency matrices are zeros. For example, consider again the graph G in Fig. 1(a). We arbitrarily and uniquely number the genes in G.

Suppose the gene array of G is [YMR043W, YJL159W, YLR131C, YGR125W, YAL051W]. Then the gene-adjacency matrix of G is

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{1}$$

Definition 6 (Core of a Gene-Adjacency Matrix). Let A be a gene-adjacency matrix with k genes (i.e., A has k rows and columns). The sub-matrix consisting of the first $k-1$ rows and columns (i.e., the first $k-1$ genes) of A is called the *core* of A.

The matrix in (2) shows the core of the matrix in (1) where the gene array of the core is [YMR043W, YJL159W, YLR131C, YGR125W].

$$\begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{2}$$

Definition 7 (Equivalence Class). Let G_1 and G_2 be two graphs with the same size. Let $A(G_1)$ ($A(G_2)$, respectively) be the gene-adjacency matrix of G_1 (G_2, respectively). G_1 and G_2 are said to be in the same *equivalence class* if $A(G_1)$ and $A(G_2)$ have the same core.

The proposed pattern discovery algorithms employ the Apriori principle [2, 22], which says that, if a pattern is frequent, then all its sub-patterns must also be frequent. The algorithms iteratively execute two steps: candidate generation and frequency counting. In the candidate generation step, we join smaller patterns that are already frequent to generate candidate patterns of larger sizes. In the frequency counting step, we check whether the support of a candidate pattern is greater than or equal to *minsup* to determine whether the candidate pattern is frequent or not. Only frequent patterns are kept. Below we explain in detail how the two steps work.

2.2 Candidate Generation and Pattern Discovery

2.2.1 Phase 1: Single-Gene Pattern Discovery

It is straightforward to find all frequent 1-patterns (i.e., frequent genes). For each gene g, we create a supporting list comprising the identifiers (ids) of the graphs that contain g. If the cardinality of the supporting list is greater than or equal to *minsup*, then g is frequent; otherwise g is not frequent.

	g_x	g_y
g_x	0	0
g_y	0	0

(a) No edge: between g_x and g_y

	g_x	g_y
g_x	0	1
g_y	0	0

(b) One edge: from g_x to g_y

	g_x	g_y
g_x	0	0
g_y	1	0

(c) One edge: from g_y to g_x

	g_x	g_y
g_x	0	1
g_y	1	0

(d) Two edges: g_x, g_y pointing to each other

Fig. 2. Four possible 2-patterns generated from frequent genes g_x, g_y.

2.2.2 Phase 2: Two-Gene Pattern Discovery

The second phase of our pattern growth algorithm is to find all frequent 2-patterns (i.e., patterns with two genes). We generate candidate 2-patterns by considering all possible pairs of the frequent genes found in phase 1. Suppose g_x and g_y are two frequent genes. Figure 2 shows four possible 2-patterns generated from g_x, g_y, in which the disconnected 2-pattern in Fig. 2(a) is eliminated from further consideration. The other three 2-patterns become candidate patterns.

For each candidate 2-pattern, its supporting list is constructed by taking the intersection of the supporting lists of the two corresponding genes. If the cardinality of the intersection list is greater than or equal to *minsup*, then we calculate the support of the candidate 2-pattern by invoking the candidate pattern verification procedure described in the following subsection; otherwise the candidate 2-pattern is discarded. Only frequent 2-patterns are kept.

2.2.3 Phase 3: $(k + 1)$-Pattern Discovery

Let P_1 and P_2 be two frequent k-patterns that are in the same equivalence class. Let A_1 (A_2, respectively) be the gene-adjacency matrix of P_1 (P_2, respectively). By definition, A_1 and A_2 have the same core. We now consider how to join P_1 and P_2 to generate candidate $(k + 1)$-patterns. Let the gene array of P_1 (P_2, respectively) be $[g_1, \ldots, g_{k-1}, g_x]$ ($[g_1, \ldots, g_{k-1}, g_y]$, respectively). Look at the last genes g_x in P_1 and g_y in P_2. There are two cases.

Case 1. We generate four candidate $(k+1)$-patterns by assigning g_x to be the last gene and g_y to be the second to the last gene to get the gene array $[g_1, \ldots, g_{k-1}, g_y, g_x]$ for the four candidate $(k+1)$-patterns. The edges of P_1 and P_2 are copied to the four candidate $(k+1)$-patterns. Then we create edges as follows. In the first $(k+1)$-pattern, there is no edge between g_x and g_y. We have to check whether there is an undirected path between g_x and g_y. If such a path does not exist, this pattern is disconnected, and hence discarded. In the second $(k+1)$-pattern, create an edge from g_x to g_y. In the third $(k+1)$-pattern, create an edge from g_y to g_x. In the fourth $(k+1)$-pattern, create an edge from g_x to g_y and an edge from g_y to g_x.

Case 2. We generate another four candidate $(k+1)$-patterns by assigning g_y to be the last gene and g_x to be the second to the last gene to get the gene array $[g_1, \ldots, g_{k-1}, g_x, g_y]$ for the four candidate $(k+1)$-patterns. Then create edges between g_x and g_y as in case 1.

2.3 Frequency Counting

Let the size of a candidate pattern P be k and the size of the largest graph $G \in \mathcal{D}$ be n. There are at most $k \times (k-1)$ edges in P and at most $n \times (n-1)$ edges in G. We pre-process the graphs in $\mathcal{D}$ by storing their nodes and edges in hash tables respectively. This pre-processing takes $O(n^2 \times |\mathcal{D}|)$ time where $|\mathcal{D}|$ represents the total number of graphs in $\mathcal{D}$. Central to our frequency counting step is the candidate pattern verification procedure, described below, for calculating the support of P. If the support is greater than or equal to *minsup*, then P is frequent and becomes a qualified pattern.

2.3.1 Induced Pattern Verification

To determine whether P is an induced pattern of a graph G, we check whether the gene array of P is a subarray of the gene array of G (step 1). If yes, then we extract the subarray from G and check whether the edges of P are identical to the edges connecting the genes in the extracted subarray of G (step 2). Step 1 takes $O(k)$ time when the node hash table is used. Step 2 takes $O(k^2)$ time. Thus, checking whether P is an induced pattern of G takes $O(k^2)$ time in total. The time complexity of checking whether P is a frequent induced pattern in $\mathcal{D}$ is $O(k^2 \times |\mathcal{D}|)$.

2.3.2 General Pattern Verification

We hash each edge of P to see whether it matches an edge of G in the edge hash table, which takes $O(k \times (k-1))$ time. Therefore, checking whether P is a general pattern of G takes $O(k^2)$ time. The time complexity of checking whether P is a frequent general pattern in $\mathcal{D}$ is also $O(k^2 \times |\mathcal{D}|)$.

2.4 Proposed Algorithms

The two proposed algorithms, IPD (induced pattern discovery) and GPD (general pattern discovery), work in a similar way. The difference is that IPD uses the induced pattern verification procedure while GPD uses the general pattern verification procedure in the frequency counting step. We describe IPD in Algorithm 1 below, omitting the description of GPD.

In Algorithm 1, FS_k contains all frequent k-patterns, and FS contains all discovered frequent patterns. At step 7, the algorithm finds all frequent $(k+1)$-patterns based on frequent k-patterns in FS_k by (i) performing pairwise join of patterns in each equivalence class in FS_k to generate all candidate $(k+1)$-patterns, and (ii) calculating the support of each candidate $(k+1)$-pattern by invoking the induced pattern verification procedure. Those candidate $(k+1)$-patterns whose support values are greater than or equal to *minsup* are frequent, and stored in FS_{k+1}.

Algorithm 1. Induced Pattern Discovery (IPD)

```
1: procedure IPD($\mathcal{D}$, $minsup$)
2:     $FS_1 \leftarrow$ {frequent 1-patterns}
3:     $FS_2 \leftarrow$ {frequent 2-patterns}
4:     $FS \leftarrow FS_1 \cup FS_2$
5:     $k \leftarrow 2$
6:     while $|FS_k| > 0$ do
7:         $FS_{k+1} \leftarrow$ {frequent $(k+1)$-patterns}
8:         $FS \leftarrow FS \cup FS_{k+1}$
9:         $k \leftarrow k+1$
10:    end while
11: return $FS$
12: end procedure
```

2.5 Correctness and Complexity Analysis

We present in this section a series of lemmas and theorems concerning the IPD algorithm. These lemmas and theorems hold for the GPD algorithm as well.

Lemma 1. *IPD is correct. That is, any pattern output from IPD is a frequent induced pattern in the given set of graphs $\mathcal{D}$.*

Proof. In order for a candidate pattern to qualify as a frequent induced pattern in $\mathcal{D}$, it has to pass the induced pattern verification procedure and satisfy the *minsup* requirement. The correctness of IPD is obvious, and hence the lemma is proved.

Lemma 2. *IPD is complete. That is, it does not miss any frequent induced pattern in the given set of graphs $\mathcal{D}$.*

Proof. We prove this lemma by mathematical introduction.

Base step. Clearly, IPD finds all frequent induced 1-patterns and 2-patterns, because it generates all such patterns using a brute force enumeration method.

Hypothesis step. Assume the lemma holds for frequent induced k-patterns; that is, all such patterns can be found by IPD.

Induction step. We want to show that IPD does not miss any frequent induced $(k+1)$-pattern. It suffices to prove that any frequent induced $(k+1)$-pattern can always be generated by two frequent induced k-patterns in some equivalence class.

Suppose P is a frequent induced $(k+1)$ pattern. Let the gene array of P be $[g_1, g_2, \ldots, g_k, g_{k+1}]$. Let P_1 (P_2, respectively) be the pattern obtained by removing the last gene g_{k+1} (the second to the last gene g_k, respectively) and edges connecting g_{k+1} (g_k, respectively) from P. Obviously, P_1 and P_2 are induced patterns. Since P is frequent, P_1 and P_2 must be frequent. By the induction hypothesis, IPD can find both P_1 and P_2. Thus, the two patterns are in FS_k. Furthermore, these two patterns are in the same equivalence class. Based on the logic of IPD, the two patterns P_1 and P_2 must be joined together by our algorithm. Since IPD exhaustively considers all possible expansions of patterns, P must be in the candidate set obtained by joining P_1 and P_2. This completes the proof.

Theorem 1. *IPD correctly finds all frequent induced patterns in the given set of graphs $\mathcal{D}$.*

Proof. From Lemmas 1 and 2 and the fact that IPD is based on a candidate generation and verification scheme, the theorem is proved.

Theorem 2. *Let M be the total number of frequent induced patterns, and K be the size of the largest frequent induced pattern in $\mathcal{D}$. The time complexity of IPD is $O(M^2 \times K^2 \times |\mathcal{D}|)$.*

Proof. Suppose there are m frequent induced k-patterns. We can generate at most $8 \times (m \times (m-1)/2)$ candidate $(k+1)$-patterns by joining two frequent induced k-patterns. Checking whether a candidate $(k+1)$-pattern is frequent takes $O(k^2 \times |\mathcal{D}|) \leq O(K^2 \times |\mathcal{D}|)$ time. There are $O(M \times (M-1)/2)$ valid pairs of joining. Therefore, the time complexity of IPD is $O(M^2 \times K^2 \times |\mathcal{D}|)$.

Notice that this is a very pessimistic upper bound for three reasons. First, the actual number of graphs involved in the verification and frequency counting phase for each candidate pattern is much smaller than $|\mathcal{D}|$. With the pattern size growing, the number of graphs that need to be checked against each pattern drops quickly. Second, the actual size of most candidate patterns that need to be verified in the frequency counting phase is smaller than K. Third, the pairwise joining operations occur only in the same equivalence class. It is less likely that all of the frequent induced k-patterns would be in the same equivalence class. Thus, the actual number of candidate patterns generated by the algorithm is much smaller than $O(M^2)$. Finally, we note that this is a pseudo-polynomial time algorithm, since M and K are not input parameters but values derived from the output [22].

3 Experiments and Results

3.1 Datasets

We used GeneNetWeaver [18] to generate the datasets related to E. coli and yeast. GeneNetWeaver has been widely used to generate benchmark datasets for evaluating GRN inference tools. Specifically we built three different GRNs for each species where the GRNs contained 10, 20, 30 genes (or nodes) respectively. Table 1 presents details of the E. coli GRNs, showing the number of nodes (edges, respectively) in each GRN. Table 2 presents details of the yeast GRNs. Each GRN is the gold standard or ground truth for the corresponding network size of each species.

Table 1. E. coli GRNs used in the experiments

GRN	Directed	#Nodes	#Edges
E. coli 10	Yes	10	9
E. coli 20	Yes	20	23
E. coli 30	Yes	30	53

Table 2. Yeast GRNs used in the experiments

GRN	Directed	#Nodes	#Edges
Yeast 10	Yes	10	10
Yeast 20	Yes	20	20
Yeast 30	Yes	30	36

For each GRN, we generated two files of gene expression data, with one file containing steady-state data and the other file containing time-series data. The steady-state data used in the experiments were knockdown data. A knockdown is a technique to reduce the expression of a gene, which is simulated by reducing the transcription rate of this gene by half. The time-series data file contained 5 time series, where each time series consisted of 21 time points.

3.2 Experimental Setup

We compared our approach of using graph mining for GRN inference with three existing GRN inference methods: GENIE3 [7], Jump3 [8] and TimeDelay-ARACNE (abbreviated as TD-ARACNE) [23]. All the three methods are well-known and widely-used for GRN inference. GENIE3 takes as input a file of steady-state gene expression data (knockdowns in our study) and produces as output a ranked list of weighted directed edges. Jump3 takes as input time series gene expression data and also produces as output a ranked list of weighted directed edges. For GENIE3 (Jump3, respectively), we select from its output $(1.5 \times n)$ top-ranked edges with the largest weights where n is the number of nodes as suggested in the literature [14]. These selected edges constitute the GRN inferred by GENIE3 (Jump3, respectively). TD-ARACNE takes as input time series gene expression data and produces a set of directed edges. All these edges constitute the GRN inferred by TD-ARACNE.

For each network size, GENIE3 infers one GRN while each of Jump3 and TD-ARACNE infers five GRNs since there are five time series generated by GeneNetWeaver. Our approach to reconstructing a GRN is to take these eleven GRNs and find frequent patterns in the eleven GRNs using the proposed IPD (GPD, respectively) algorithm with $minsup = 4$. The edges in the found patterns constitute the GRN inferred by IPD (GPD, respectively).

For an inferred GRN G, a true positive is an edge in G, which is also an edge in the gold standard. A false positive is an edge in G, which, however, is not an edge in the gold standard. A true negative is an edge not in G, which is not an edge in the gold standard either. A false negative is an edge not in G, which, however, is an edge in the gold standard.

Let TP (FP, TN, FN, respectively) denote the number of true positives (false positives, true negatives, false negatives, respectively). P is the sum of TP and FN. N is the sum of TN and FP. We use accuracy (ACC), defined below, to evaluate the performance of a GRN inference method.

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}} \tag{3}$$

3.3 Experimental Results

Figure 3 shows the accuracies of the five GRN inference methods studied in the paper for varying E. coli dataset sizes. The accuracy of Jump3 (TD-ARACNE, respectively) was calculated by taking the average of the accuracies computed using the GRNs inferred from the five time series generated by GeneNetWeaver. It can be seen from Fig. 3 that the proposed IPD and GPD algorithms outperform the three existing GRN inference methods, producing more accurate GRNs than the existing methods. This happens probably because the frequent patterns found by the proposed algorithms are more likely to occur in the ground truth. Figure 4 shows the accuracies of the five GRN inference methods for varying yeast dataset sizes. The results for yeast are consistent with those for E. coli, showing that the proposed algorithms are the best among the five methods. The figures indicate that our approach is capable of predicting highly reliable edges in a GRN.

In general, IPD is slightly better than GPD in GRN inference. Both IPD and GPD employ a *minsup* parameter. A large *minsup* value would yield very few patterns while a small *minsup* value would yield too many patterns. For both cases, the numbers of edges in the GRNs constructed by our proposed algorithms would deviate from those of the GRNs inferred by the three existing methods. As a consequence, we set *minsup* to 4, which yielded the best results.

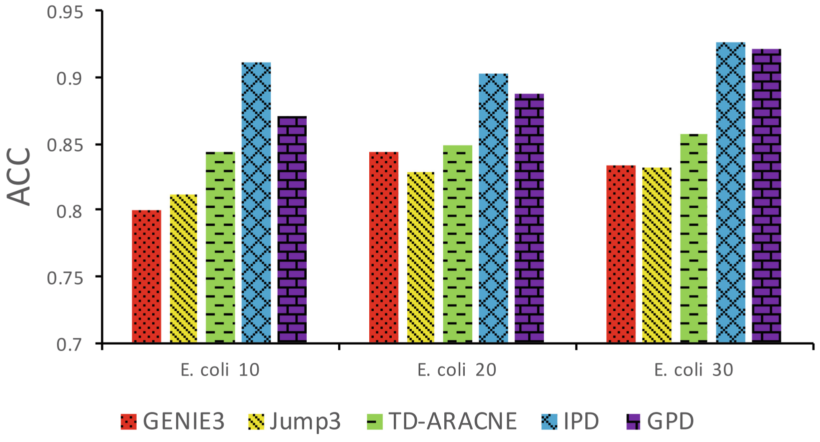


Fig. 3. Accuracies of the five GRN inference methods studied in the paper on E. coli datasets.

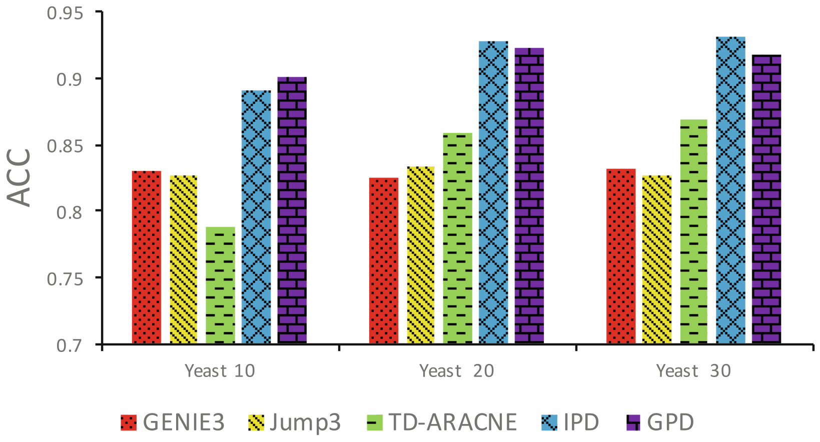


Fig. 4. Accuracies of the five GRN inference methods studied in the paper on yeast datasets.

4 Conclusions

In this paper we present a graph mining approach to reverse engineering gene regulatory networks. Our graph mining algorithms are capable of discovering frequent or common patterns from the GRNs inferred by existing methods. These common patterns are more likely to occur in true regulatory networks. Our experimental results demonstrated the good performance of the proposed approach.

We considered here three existing GRN inference methods, namely GENIE3 [7], Jump3 [8] and TimeDelay-ARACNE [23]. In the future, we plan to explore additional GRN inference methods, evaluating the feasibility of our approach for those methods. We also plan to extend the techniques presented here to miRNA-mediated gene regulatory networks. We have recently studied how microRNAs (miRNAs) can regulate gene expression and alternative polyadenylation through their interaction at the 3′ untranslated region (UTR) of mammalian mRNAs, especially in genes of inflammatory pathways [12,19]. We plan to use graph mining methods to analyze these pathways.

References

1. Abduallah, Y., Turki, T., Byron, K., Du, Z., Cervantes-Cervantes, M., Wang, J.T.L.: Mapreduce algorithms for inferring gene regulatory networks from time-series microarray data using an information-theoretic approach. BioMed. Res. Int. **2017**, 6261802 (2017)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Data Bases, pp. 487–499 (1994)
3. Angelini, C., Costa, V.: Understanding gene regulatory mechanisms by integrating ChIP-seq and RNA-seq data: statistical solutions to biological problems. Front. Cell Dev. Biol. **2**, 51 (2014)
4. Brazhnik, P., de la Fuente, A., Mendes, P.: Gene networks: how to put the function in genomics. Trends Biotechnol. **20**(11), 467–472 (2002)
5. Elloumi, M., Iliopoulos, C., Wang, J.T.L., Zomaya, A.Y.: Pattern Recognition in Computational Molecular Biology: Techniques and Approaches. Wiley, Hoboken (2016)
6. Huan, J., Wang, W., Prins, J.: Efficient mining of frequent subgraphs in the presence of isomorphism. In: Proceedings of the 3rd IEEE International Conference on Data Mining, pp. 549–552 (2003)
7. Huynh-Thu, V.A., Irrthum, A., Wehenkel, L., Geurts, P.: Inferring regulatory networks from expression data using tree-based methods. PLoS ONE **5**(9), e12776 (2010)
8. Huynh-Thu, V.A., Sanguinetti, G.: Combining tree-based and dynamical systems for the inference of gene regulatory networks. Bioinformatics **31**(10), 1614–1622 (2015)
9. Ideker, T., Krogan, N.J.: Differential network biology. Mol. Syst. Biol. **8**(1), 565 (2012)
10. Inokuchi, A., Washio, T., Motoda, H.: Complete mining of frequent patterns from graphs: mining graph data. Mach. Learn. **50**(3), 321–354 (2003)
11. Jiang, C., Coenen, F., Zito, M.: A survey of frequent subgraph mining algorithms. Knowl. Eng. Rev. **28**(1), 75–105 (2013)
12. Kasar, S., Underbayev, C., Hassan, M., Ilev, I., Degheidy, H., Bauer, S., Marti, G., Lutz, C.S., Raveche, E., Batish, M.: Alterations in the mir-15a/16-1 loci impairs its processing and augments B-1 expansion in de novo mouse model of chronic lymphocytic leukemia (CLL). PLoS ONE **11**(3), e0149331 (2016)
13. Kuramochi, M., Karypis, G.: Finding frequent patterns in a large sparse graph. In: Proceedings of the SIAM International Conference on Data Mining (2004)
14. Leclerc, R.D.: Survival of the sparsest: robust gene networks are parsimonious. Mol. Syst. Biol. **4**, 213 (2008)
15. Margolin, A.A., Wang, K., Lim, W.K., Kustagi, M., Nemenman, I., Califano, A.: Reverse engineering cellular networks. Nat. Protoc. **1**(2), 662–671 (2006)

16. Nijssen, S., Kok, J.N.: The Gaston tool for frequent subgraph mining. Electron. Notes Theoret. Comput. Sci. **127**(1), 77–87 (2005)
17. Patel, N., Wang, J.T.L.: Semi-supervised prediction of gene regulatory networks using machine learning algorithms. J. Biosci. **40**(4), 731–740 (2015)
18. Schaffter, T., Marbach, D., Floreano, D.: GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. Bioinformatics **27**(16), 2263–2270 (2011)
19. Shukla, S., Elson, G., Blackshear, P.J., Lutz, C.S., Leibovich, S.J.: 3'UTR AU-rich elements (AREs) and the RNA-binding protein Tristetraprolin (TTP) are not required for the LPS-mediated destabilization of phospholipase-C beta-2 mRNA in murine macrophages. Inflammation **40**(2), 645–656 (2017)
20. Werner, T., Dombrowski, S.M., Zgheib, C., Zouein, F.A., Keen, H.L., Kurdi, M., Booz, G.W.: Elucidating functional context within microarray data by integrated transcription factor-focused gene-interaction and regulatory network analysis. Eur. Cytokine Netw. **24**(2), 75–90 (2013)
21. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: Proceedings of the 2002 IEEE International Conference on Data Mining (2002)
22. Zhang, S., Wang, J.T.L.: Discovering frequent agreement subtrees from phylogenetic data. IEEE Trans. Knowl. Data Eng. **20**(1), 68–82 (2008)
23. Zoppoli, P., Morganella, S., Ceccarelli, M.: TimeDelay-ARACNE: reverse engineering of gene networks from time-course data by an information theoretic approach. BMC Bioinform. **11**(1), 154 (2010)

Spammer Detection via Combined Neural Network

Weiping Pei[1,2], Youye Xie[2], and Gongguo Tang[2](✉)

[1] Biliang, Ltd., Zhuhai, China
[2] Colorado School of Mines, Golden, USA
{weipingpei,youyexie,gtang}@mines.edu

Abstract. Social networks, as an indispensable part of our daily lives, provide ideal platforms for entertainment and communication. However, the appearance of spammers who spread malicious information pollutes a network's reliability. Unlike email spammers detection, a social network account has several types of attributes and complicated behavior patterns, which require a more sophisticated detection mechanism. To address the above challenges, we propose several efficient profiles and behavioral features to describe a social network account and a combined neural network to detect the spammers. The combined neural network can process the features separately based on their mutual correlation and handle data with missing features. In experiments, the combined neural network outperforms several classical machine learning approaches and achieves 97.5% accuracy on real data. The proposed features and the combined neural network have already been applied commercially.

Keywords: Spammer detection · Social network · Deep learning
Data mining

1 Introduction

1.1 Background

The development of social network platforms, such as Twitter, Facebook, and Sina, have made communication around the world much more convenient. With the increasing impact of the social network, a significant number of spammers appears aiming to conduct malicious behaviors, such as spreading malicious URLs, posting abusive comments and hijacking the social hotspot. And that makes spammers detection one of the top priorities of social network companies. This paper focuses on spammers detection for the Sina microblogs, one of the most influential social network platforms in China, which allows users to post microblogs with less than 140 characters along with images or videos. The spammers are also called the paid posters or internet water army on Sina [3]. The analysis on feature effectiveness and the proposed combined neural network are

W. Pei and Y. Xie have contributed equally to this work.

https://doi.org/10.1007/978-3-319-96136-1_28

also applicable to other social networks such as Twitter and Facebook by simply changing the language analysis tool and modifying the number of input nodes accordingly.

1.2 Related Work

Spammers detection appears first in email applications, the goal of which is to filter out spam emails based on the email content [6,12], the abnormal behavior [10,17] and attachments. Social network companies such as Twitter and Sina face similar issues and many methods have been developed for spammers detection in social networks. O'Donovan et al. [11] analyzed the content features of social network accounts and suggested the usefulness of content features in spammers detection. As a complement, Liu et al. [9] studied the behavioral features and proposed a hybrid model to calculate the 'spamming value' for each account.

In addition, machine learning approaches have also produced promising results for spammers detection. Wang [15] proposed a Naïve Bayes method using three graph-based features (profile features) and three content features. Two support vector machine (SVM) based methods were developed by Cheng et al. [5] and Zheng et al. [18], whose approach detects spammers in two phases. The first phase utilizes content features and the second phase considers the topic of the microblogs with the help of a Latent Dirichlet Allocation (LDA) model [1]. Chakraborty et al. [2] designed a Social Profile Abuse Monitoring (SPAM) system which also adopts the multi-stages detection mechanism and achieves 89% accuracy.

However, most of the previous approaches assume that the account has all the features the detection model requires, which is not desirable in practice. Indeed, a new spammer account may not have any significant behavioral features at all. In addition, some proposed features, such as the content topic recognition and accounts similarity analysis, require more feature engineering effort. Our work contribute to the topic of spammers detection in the following ways:

- **Efficient profile and behavioral features.** We investigate the difference between spammers and non-spammers and propose the use of several profile and behavioral features that can be extracted easily from the social network accounts.
- **Combined neural network.** We propose a novel combined neural network that includes a linear regression model (LR) and two artificial neural networks (ANN) to incorporate different types of features. More importantly, the combined model is very flexible in practice since each sub-model within the combined neural network can perform the detection independently, by simply deactivating unwanted sub-models, in the case that some features are missing.
- **High detection accuracy.** We conduct experiments to study the effectiveness of the proposed features and the detection performance of the combined neural network and its sub-models. In our experiment, the combined neural network outperforms other classical machine learning models in literature by achieving 97.5% detection accuracy.

The paper is organized as follows. Section 2 introduces the profile and behavioral features that will be used for our detection model. In Sect. 3, we present our combined neural network and derive its training process. We study the effectiveness of the proposed network in spammers detection in Sect. 4. The paper is concluded in Sect. 5.

2 Features

A social network contains massive data that may change rapidly. So extracting distinctive features is a critical component for spammers detection. Based on the features categories, we consider four profile features and four behavioral features for our model and we will analyze those features' effectiveness in experiments. Some of the features in this paper have proven efficiency in the spammer detection problem [2,5,11,15].

2.1 Profile Features

A registered account in a social network typically has a public profile which describes the attributes of the user. The normal users are more likely to fill out the basic profile so that their friends could recognize them. However, it is cumbersome for spammers to create a lot of accounts with complete profile information. This section analyzes four profile features that are useful in spammers detection and summarizes them in Table 1.

- **Fans ratio.** The followers, also called fans, are users that follow one's account, while the followings are the users that one follows. Spammers would follow many normal users in the hope that they would follow back so that they can spread spam messages. And a few normal users would indeed follow a stranger. This results in a limited number of followers and lots of followings for spammers. We define *fans_ratio* to quantify this phenomenon

$$fans\_ratio = \frac{follower}{follower + following}$$

  where *follower* and *following* represent the number of followers and followings respectively. We collect 1000 spammers and non-spammers' *fans_ratio* in Fig. 1. It is obvious that spammers (red cross) are more likely to have lower *fans_ratio* than non-spammers (green cross).
- **Account level.** Each account has a level indicator that reflects the activeness of this account, determined by how long the account is established and how often the user posts a microblog. Normally, an active account would have a higher level than a silent one. We record 1000 spammers and non-spammers' *account levels* in Fig. 2. To compensate the level data's disproportion, all the levels are normalized within $[0, 1]$ for training and testing.
- **Verification.** The Sina microblog network enforces an identification policy. An account can earn a verification mark by verifying its owner's identity. Since detailed personal information are needed for verification, few spammers have such a verification mark.

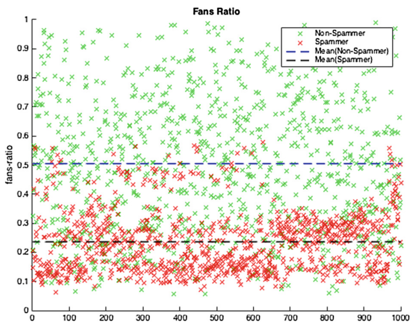


Fig. 1. The *fans_ratio* feature. (Color figure online)

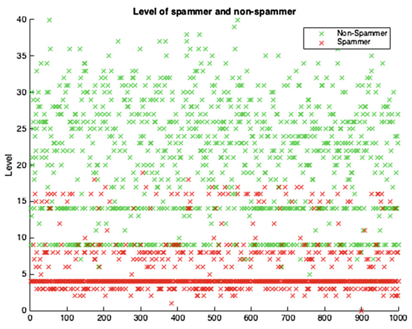


Fig. 2. The *account level* feature. (Color figure online)

Table 1. The proposed profile features.

Feature name	Definition
fans_ratio	The fans ratio
lv	The account level normalized within $[0, 1]$
is_common	The account is not verified
is_V	The account is verified as a personal account
is_expert	The account is verified as an expert's account
is_company	The account is verified as an organization's account
Brief information	The account has an introduction

– **Self introduction.** A brief self introduction can be added for each account that is displayed under the account name. Most spammers do not have a self introduction.

2.2 Behavioral Features

Besides the profile features, spammers' accounts also operate in a different manner due to their specific blogging purposes. Behavioral features provide a way to quantify that difference. We propose four behavioral features which can be easily extracted from the microblogs of an account.

– **User interaction.** For each microblog, other users can comment, repost it or leave a'like'. The microblog itself might also be a reposted one. The number of interaction activities reflects the attention a microblog has attracted. We binarize those features based on whether a particular microblog has the above interaction activities.
– **Special characters.** The microblog also provides other enhanced interaction using special characters. We summarize those special characters and their associated functions in Table 2.

Table 2. The special characters and proposed content features.

Special characters	Function
I	The subject of the message
@	Post a microblog and alert other users
#	Post a microblog with hashtags
//@	Repost others reposted microblogs
URLs	Attach the URLs
Video	Attach the video
Content features	Definition
Number of words	The number of words normalized within $[0, 1]$
Non-repeated words	Fraction of non-repeated words
Non-stop words	Fraction of non-stop words
Non-repeated and non-stop words	Fraction of non-repeated and non-stop words

– **Content features.** Instead of doing the content mining, this paper leverages simpler content features in terms of the text length and the valuable content. We segment the microblog's content and compute the corresponding content features as summarized in Table 2. Generally, stop words are a group of words that occur frequently but do not carry much information, like the word 'the' in English.
– **Publish time.** Controlling by the software, spammers keep posting even at deep night and the interval time between two microblogs is more regular compared to non-spammers. Therefore, we use an 8×1 vector, whose i-th entry is 1 when the microblog is posted during $[(i-1) \times 3:00 - (i \times 3):00]$, to indicate the publish time. A 7×1 vector is used to represent the day of the week of the posting.

3 The Combined Neural Network

In the past decade, neural networks have achieved great success in many machine learning tasks [8,13,16]. A properly designed neural network model has sufficient capacity to classify complicated data in high dimensional spaces. It is also found that hybrid models which utilize multiple models in a proper way can improve the classification performance [4]. However, they fail to consider the situation of missing data. Inspired by those ideas, we generalize and propose a combined neural network which hybridizes linear regression models and artificial neural networks. Specifically, for the spammer detection problem, our combined network is composed of one linear regression model (LR) and two neural networks (ANN) as shown in Fig. 3.

With the proposed combined network, highly correlated features can be processed together in a sub-networks to avoid the influence of other less related features. In practice, if some of the account's features are absent, the sub-models

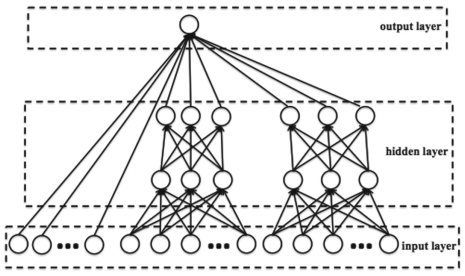


Fig. 3. Architecture of the proposed combined neural network.

Table 3. The combined neural network for spammers detection. The structures show the number of nodes from input to output layers for each sub-model.

Submodel	Feature name	Structures
LR	Profile features	[7 2]
ANN-1	Publish time	[150 150 52 2]
ANN-2	User interaction, characters and content features	[140 140 49 2]

can still perform the detection independently. In addition, the design reduces computational complexity compared to a fully connected neural network and makes it easier to handle inputs with different types of attributes.

Based on the features analysis in the last section, we category the features associated with an account into three classes, the profile features, the publish time and the behavioral features (except the publish time) whose dimensions are $7, 150$ (15×10 for ten most recent microblogs) and 140 (14×10 for ten most recent microblogs) respectively. Therefore, given N undefined accounts, the network input is $\mathbf{X} = [\mathbf{X}^L, \mathbf{X}^{A_1}, \mathbf{X}^{A_2}]^T \in \mathbf{R}^{297 \times N}$ where $\mathbf{X}^L \in \mathbf{R}^{7 \times N}$, $\mathbf{X}^{A_1} \in \mathbf{R}^{150 \times N}$ and $\mathbf{X}^{A_2} \in \mathbf{R}^{140 \times N}$.

We model the profile features using linear regression (i.e. a neural network without hidden layers) since those features are almost linear separable based on the experiments. The rest two classes are modeled using two independent artificial neural networks due to their non-linear characteristics. Empirically, we observe that when the numbers of nodes in the first and second hidden layers equals the 100% and 35% of the number of nodes in the input layer, the combined model can achieve best performance as shown in Table 3.

3.1 Model Details

We first introduce parameters' notations for the combined neural network in Table 4. The corresponding matrices are written in boldface capital letters without subscripts. For example, $\mathbf{W}^{L,1}$ represents the weight matrix connecting the first and second layer in the linear model. In addition, since the input layer is the first layer, we have $x_i^L = a_i^{L,1}$ and $x_i^A = a_i^{A,1}$.

Table 4. The combined neural network's parameters.

Notation	Definition
y_i	The i-th node's output in the output layer
$w_{ij}^{L,l}, w_{ij}^{A,l}$	The weight parameters connecting the i-th node in the l layer to the j-th node in the $(l+1)$ layer in the linear model and ANN
$b_j^{L,l}, b_j^{A,l}$	The bias parameters connecting to the j-th node of the $(l+1)$ layer in the linear model and ANN
$z_i^{L,l}, z_i^{A,l}$	The i-th node's input of the l-th layer in the linear model and ANN
$a_i^{L,l}, a_i^{A,l}$	The i-th node's output of the l-th layer in the linear model and ANN

The linear model can be easily shown as a linear transformation $\mathbf{Z}^{L,2} = \mathbf{W}^{L,1}\mathbf{X}^L + \mathbf{B}^{L,1}$. Similarly, for the ANN, we have $\mathbf{Z}^{A,4} = \mathbf{W}^{A,3}f(\mathbf{W}^{A,2}f(\mathbf{W}^{A,1}\mathbf{X}^A + \mathbf{B}^{A,1}) + \mathbf{B}^{A,2}) + \mathbf{B}^{A,3}$ where $f(x) = \max(0, x)$ is the rectifier activation function. Based on the above equations, the output layer's input for the combined neural network is $\mathbf{Z} = \mathbf{Z}^{L,2} + \mathbf{Z}^{A_1,4} + \mathbf{Z}^{A_2,4}$.

A softmax function is applied in the output layer to calculate the posterior probabilities. For one particular data point, the network's output is

$$\mathbf{Y} = \left[\frac{\exp\left(z_1^{L,2} + z_1^{A_1,4} + z_1^{A_2,4}\right)}{\sum_{i=1}^{2}\exp\left(z_i^{L,2} + z_i^{A_1,4} + z_i^{A_2,4}\right)}, \frac{\exp\left(z_2^{L,2} + z_2^{A_1,4} + z_2^{A_2,4}\right)}{\sum_{i=1}^{2}\exp\left(z_i^{L,2} + z_i^{A_1,4} + z_i^{A_2,4}\right)}\right]^T$$

where $y_1 = P(\text{Spammer}|x)$ and $y_2 = P(\text{Normal}|x)$. If $y_1 \geq y_2$, the account will be classified as a spammer.

3.2 Model Training

We first derive the parameters derivatives for training. Assuming we have N data points, we evaluate the combined neural network's performance using the cross entropy

$$\mathbf{H} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{2} y_i^{'(n)} \log(y_i^{(n)})$$

where y_i' takes a value either 0 or 1 indicating the ground truth label and the minimum value of $\mathbf{H}$ is 0. Therefore, given N data points, the derivatives of the linear model parameters are:

$$\begin{aligned}\nabla\mathbf{H}(w_{ij}^{L,1}) &= \frac{1}{N}\sum_{n=1}^{N}\nabla\mathbf{H}(w_{ij}^{L,1},x^{(n)},y^{(n)}) = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{2}\frac{\partial\mathbf{H}}{\partial z_k^{L,2,(n)}}\frac{\partial z_k^{L,2,(n)}}{\partial w_{ij}^{L,1}}\\ &= \frac{1}{N}\sum_{n=1}^{N}(y_j^{(n)}-y_j^{',(n)})a_i^{L,1,(n)}\\ \nabla\mathbf{H}(b_j^{L,1}) &= \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{2}\frac{\partial\mathbf{H}}{\partial z_k^{L,2,(n)}}\frac{\partial z_k^{L,2,(n)}}{\partial b_j^{L,1}} = \frac{1}{N}\sum_{n=1}^{N}(y_j^{(n)}-y_j^{',(n)})\end{aligned}$$

in which

$$\begin{aligned}\frac{\partial\mathbf{H}}{\partial z_k^{L,2}} &= \sum_{j=1}^{2}\frac{\partial\mathbf{H}}{\partial y_j}\frac{\partial y_j}{\partial z_k^{L,2}} = -y'_k + y_k\sum_{j=1}^{2}y'_j = y_k - y'_k\\ \frac{\partial z_k^{L,2}}{\partial w_{ij}^{L,1}} &= \begin{cases}a_i^{L,1} & k=j\\ 0 & k\neq j\end{cases}, \quad \frac{\partial z_k^{L,2}}{\partial b_j^{L,1}} = \begin{cases}1 & k=j\\ 0 & k\neq j.\end{cases}\end{aligned}$$

The derivatives of the ANN can be calculated efficiently via backpropagation. We first compute the derivative with respect to each node's input, denoted as δ, and propagate it backward. For the output layer and the l-th ($l \leq 3$) layer

$$\delta_i^4 = \frac{\partial\mathbf{H}}{\partial z_i^{A,4}} = y_i - y'_i, \quad \delta_i^l = \frac{\partial\mathbf{H}}{\partial z_i^{A,l}} = \sum_{j=1}^{n_{l+1}}\delta_j^{l+1}w_{ij}^{A,l}f'(z_i^{A,l})$$

where n_{l+1} is the total number of nodes in the $(l+1)$ layer (except the bias node) and $f'(z_i^{A,l})$ is the derivative of the rectifier function. $f'(z_i^{A,l}) = 1$ if $z_i^{A,l} \geq 0$ and $f'(z_i^{A,l}) = 0$ otherwise. Therefore, we get the derivatives with respect to the weights and biases for ANN with N input data points as follows.

$$\begin{aligned}\nabla\mathbf{H}(w_{ij}^{A,l}) &= \frac{1}{N}\sum_{n=1}^{N}\nabla\mathbf{H}(w_{ij}^{A,l},x^{(n)},y^{(n)}) = \frac{1}{N}\sum_{n=1}^{N}\delta_j^{l+1,(n)}a_i^{A,l,(n)}\\ \nabla\mathbf{H}(b_j^{A,l}) &= \frac{1}{N}\sum_{n=1}^{N}\delta_j^{l+1,(n)}\end{aligned}$$

After calculating the derivatives, the Adam optimization algorithm [7] which updates the learning rate for each parameter adaptively with 0.05 initial learning rate is applied to train the combined network. Given N training samples, the batch size and number of epoches are $N/10$ and 1000 respectively. Dropout [14] is also performed in the ANN during training to prevent overfitting. Nodes in the first and second hidden layers are kept with probabilities 0.5 and 0.8.

4 Experiment and Analysis

We analyze the effectiveness of the selected features and compare the proposed combined neural network to other classical machine learning methods for spammers detection. Throughout the experiment, five-fold cross-validation is applied for a more accurate performance estimate.

4.1 Data and Evaluation Metrics

We collect about 5000 spammer accounts and 5000 non-spammer accounts through a media company and web crawling. For each account, we label it manually and extract its profile features and behavioral features accordingly. We evaluate the model performance based on four evaluation metrics, the accuracy, precision, recall and F1 score defined as follows (Table 5).

Table 5. The confusion matrix entries.

	Spammer	Non-spammer
Predicted spammer	True positive (TP)	False positive (FP)
Predicted non-spammer	False negative (FN)	True negative (TN)

- **Accuracy:** the ratio of the number of correctly classified accounts over the total number of accounts. $A = \frac{TP+TN}{TP+TN+FP+FN}$.
- **Precision:** the ratio of the number of correctly classified spammers to the number of accounts that are classified as spammers. $P = \frac{TP}{TP+FP}$.
- **Recall:** the ratio of the number of correctly classified spammers to the number of spammers. $R = \frac{TP}{TP+FN}$.
- **F1 score:** a measure to examine the test accuracy and is computed as $F1 = \frac{2\times P\times R}{P+R}$.

4.2 Features Effectiveness Analysis

Single Feature. In order to verify the proposed features' effectiveness, we apply four classical machine learning classifiers for spammers detection using different single features. Those classifiers, which are also popular in literature, are C4.5, classification and regression trees (CART), support vector machine (SVM) and the Naïve Bayes (NB) classifier. Considering that both the special characters and content features are extracted from texts, we also combine them as the text feature for evaluation. The detection accuracies are summarized in Table 6. We observe that the text features and the publish time is quite distinguishable between spammers and non-spammers and except the content feature, all features achieve at least 78.92% accuracy for four classifiers.

Combined Features. Instead of using only one type of features as the input, in this section, we try different combinations among features and record the detection accuracies in Table 7. We can observe that using combined features promotes the detection accuracy significantly compared to using the single feature. In addition, no matter which model we use, utilizing all features for detection achieves the highest accuracy. Notably, SVM using all features achieves 96.53% accuracy which proves the effectiveness of the proposed features.

Table 6. The detection accuracies using a single feature.

Feature name	C4.5	CART	SVM	NB
Profile features	85.40%	85.74%	85.52%	78.92%
User interaction	84.06%	83.96%	84.77%	85.65%
Special character	86.87%	86.41%	87.70%	88.96%
Content feature	74.55%	75.07%	73.91%	66.37%
Text features	85.87%	86.44%	89.19%	89.80%
Publish time	87.21%	86.93%	86.70%	86.61%

Table 7. The detection accuracies using combined features.

Features name	C4.5	CART	SVM	NB
Profile features and user interaction	88.95%	87.97%	89.21%	86.30%
Profile and text features	92.97%	93.39%	94.71%	90.36%
Profile features and publish time	92.49%	91.95%	90.89%	88.40%
User interaction and text features	91.70%	91.74%	95.81%	95.39%
User interaction and publish time	89.76%	90.20%	92.78%	91.63%
Text features and publish time	89.25%	89.35%	93.69%	91.18%
All features	94.55%	94.06%	96.53%	96.02%

Different Lengths of Behavioral Features. Furthermore, we study the influence of different lengths of behavioral features. For each account, we use its profile features as the initial state (horizontal axis = 1) and add its recent microblogs one by one from which we extract the account's behavioral features. The results are shown in Fig. 4. After adding five microblogs' behavioral features, the accuracies of SVM and NB tend to be steady while the accuracies of C4.5 and CART fluctuate within a narrow range. In order to obtain a higher accuracy while avoiding heavy features engineering effort, using behavioral features from the ten most recent posts are most favorable.

4.3 The Combined Neural Network

In this section, we conduct experiments to evaluate the detection performance of the proposed combined model and its sub-models.

Comparison to Classicial Models. We first compare the detection performance between the proposed combined neural network and other classical

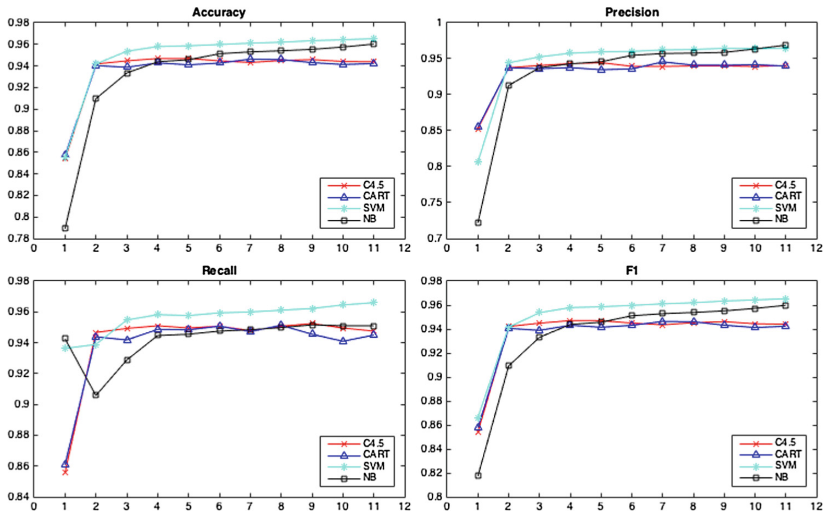


Fig. 4. The accuracy, precision, recall and F1 score with all profile features while different lengths of behavioral features.

Table 8. The detection performances for different models.

Model	Accuracy	Precision	Recall	F1 score
C4.5	94.55%	94.15%	95.01%	94.58%
CART	94.06%	93.82%	94.35%	94.08%
SVM	96.53%	96.46%	96.60%	96.53%
NB	96.02%	96.89%	95.10%	95.98%
Combined model	97.50%	98.24%	97.12%	97.68%

machine learning approaches. The comparison result is shown in Table 8 which shows that the proposed combined model achieves 97.5% accuracy and outperforms other classical machine learning classifiers in the spammer detection problem.

Sub-models Detection Performance. As we mentioned previously, each sub-model in the combined neural network can work independently in the case some features are absent. In this section, we train the combined network and extract each individual sub-model to examine its detection performance. The result is shown in Table 9.

The vote model is a multi-classifiers model that classifies an account based on the voting from three sub-models. We can find that all sub-models achieve promising detection accuracies between 84% and 97%. Therefore, it is not surprising that a vote model can achieve above 97% accuracy. However, a simple vote model ignores the relative magnitudes of each sub-model's output which

Table 9. The detection result using sub-models independently.

Model	Accuracy	Precision	Recall	F1 score
LR	84.50%	78.73%	94.58%	85.93%
ANN-1	95.68%	96.95%	75.68%	95.62%
ANN-2	96.36%	96.72%	95.97%	96.34%
Vote model	97.25%	97.80%	97.09%	97.44%
Combined model	97.50%	98.24%	97.12%	97.68%

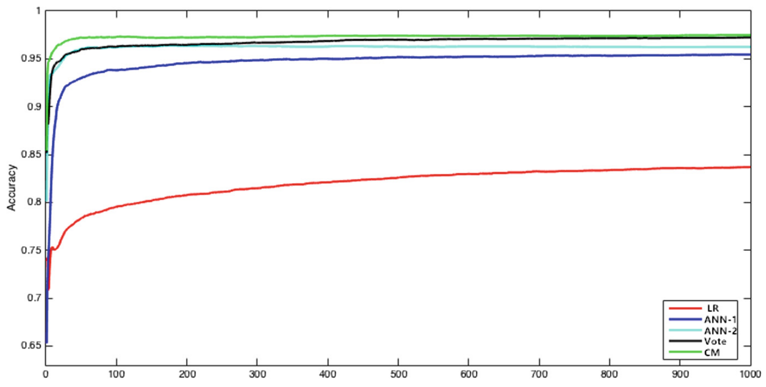


Fig. 5. Different models' accuracies versus the training iteration number.

limits its capability. We also record the sub-models training process in Fig. 5. We can observe that the combined model converges as fast as its sub-models but achieves higher accuracy. Noticeably, the linear model accuracy is far below the non-linear models due to the non-linear characteristics in the spammer detection task.

4.4 Performance Tracking

Although behavioral features are quite effective, it may update rapidly for active users. Spammers would also change their behavior pattern to escape from being detected by the platform. Therefore, it is important to examine how the fast changing environment affects the combined neural network's detection performance.

A Quick Test. We start the study from a quick test by first selecting 500 active accounts including 250 spammers and 250 non-spammers on December 25th, 2016 and implementing different classifiers on those accounts. After two months, we extracted those accounts' features again and their ten most recent microblogs were totally different from the previous. We then performed the classification again using the pre-trained models and recorded the detection accuracies in Table 10. All models' accuracies decrease after two months since those users may have already changed their microblog habits and so do the spammers. Fortunately, the combined model still achieve 96.21% accuracy and outperform other models.

Table 10. The accuracies after two months.

Date	C4.5	CART	SVM	NB	Combined model
12/25/16	95.99%	96.65%	98.88%	92.20%	99.11%
02/25/17	91.21%	87.04%	94.98%	91.63%	96.21%

Model Update. In this part, we take a closer look of the influence by the changing behavioral features. A cooperative company provides us with new accounts every day which allows us to update the combined model daily. Namely, we update the combined model using yesterday's data and test it on today's new coming data. The tracking of detection accuracies is shown in Table 11. Compared to the non-updated model, updated model performs much better and the accuracies are always beyond 97%. Therefore, it is necessary to update the model frequently.

Table 11. Accuracies tracking for the combined neural network.

Data collection date	Non-updated model	Updated model
02/19/17	93.21%	–
02/20/17	94.83%	97.24%
02/21/17	95.17%	97.24%
02/22/17	96.79%	98.57%
02/23/17	94.48%	97.59%
02/24/17	96.55%	97.24%
02/25/17	94.07%	98.15%

5 Conclusion

This paper proposes several efficient profile and behavioral features and a novel combined neural network for spammers detection in the social network. Among the massive information, our proposed features provide a decent norm to detect the spammers among legitimate users. The effectiveness of the proposed features is studied using several classical machine learning approaches. In addition, based on the correlation between different features, the combined neural network is proposed to handle the input with different types of attributes. The experiments on real world data demonstrate the efficiency and effectiveness of the proposed network which achieves 97.5% detection accuracy. Finally, we study how the combined neural network is affected by the rapidly changing internet environment.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**(Jan), 993–1022 (2003)
2. Chakraborty, A., Sundi, J., Satapathy, S., et al.: SPAM: a framework for social profile abuse monitoring. CSE508 report, Stony Brook University, Stony Brook (2012)
3. Chen, C., Wu, K., Srinivasan, V., Zhang, X.: Battling the internet water army: detection of hidden paid posters. In: 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 116–120. IEEE (2013)
4. Cheng, H.T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., et al.: Wide & deep learning for recommender systems. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, pp. 7–10. ACM (2016)
5. Cheng, Z., Kai, N., Zhiqiang, H.: Dynamic detection of spammers in Weibo. In: 2014 4th IEEE International Conference on Network Infrastructure and Digital Content (IC-NIDC), pp. 112–116. IEEE (2014)
6. Khan, A., Baharudin, B., Lee, L.H., Khan, K.: A review of machine learning algorithms for text-documents classification. J. Adv. Inf. Technol. **1**(1), 4–20 (2010)
7. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
8. 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)
9. Liu, Y., Wu, B., Wang, B., Li, G.: SDHM: a hybrid model for spammer detection in Weibo. In: 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 942–947. IEEE (2014)
10. Ming, L., Yunchun, L., Wei, L.: Spam filtering by stages. In: International Conference on Convergence Information Technology, pp. 2209–2213. IEEE (2007)
11. O'Donovan, J., Kang, B., Meyer, G., Hollerer, T., Adalii, S.: Credibility in context: an analysis of feature distributions in Twitter. In: 2012 International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2012 International Conference on Social Computing (SocialCom), pp. 293–301. IEEE (2012)
12. Ruan, G., Tan, Y.: A three-layer back-propagation neural network for spam detection using artificial immune concentration. Soft. Comput. **14**(2), 139–150 (2010)
13. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al.: Mastering the game of go with deep neural networks and tree search. Nature **529**(7587), 484–489 (2016)
14. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
15. Wang, A.H.: Detecting spam bots in online social networking sites: a machine learning approach. In: Foresti, S., Jajodia, S. (eds.) DBSec 2010. LNCS, vol. 6166, pp. 335–342. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13739-6_25
16. Xie, Y., Tang, G., Hoff, W.: Chess piece recognition using oriented chamfer matching with a comparison to CNN. In: IEEE Winter Conference on Applications of Computer Vision (WACV) (2018)

17. Yeh, C.Y., Wu, C.H., Doong, S.H.: Effective spam classification based on metaheuristics. In: 2005 IEEE International Conference on Systems, Man and Cybernetics, vol. 4, pp. 3872–3877. IEEE (2005)
18. Zheng, X., Wang, J., Jie, F., Li, L.: Two phase based spammer detection in Weibo. In: 2015 IEEE International Conference on Data Mining Workshop (ICDMW), pp. 932–939. IEEE (2015)

Pedestrian Detection: Performance Comparison Using Multiple Convolutional Neural Networks

Meenu Ajith(✉) and Aswathy Rajendra Kurup(✉)

University of New Mexico, Albuquerque, NM, USA
{majith,aswathyrk}@unm.edu

Abstract. Pedestrian Detection in real world crowded areas is still one of the challenging categories in object detection problems. Various modern detection architectures such as Faster R-CNN, R-FCN and SSD has been analyzed based on speed and accuracy measurements. These models can detect multiple objects with overlaps and localize them using a bounding box framing it. Evaluation of performance parameters provides high speed models which can work on live stream applications in mobile devices or high accurate models which provide state-of-the-art performance for various detection problems. These convolutional neural network models are tested on the Penn-Fudan Dataset as well as Google images with occlusions, which achieves high detection accuracies on each of the detectors.

1 Introduction

Pedestrian detection has made immense progress over the last few years with the arrival of convolutional neural networks. It persists as one of the challenging problems because of the large variability of pedestrians in clothing, so that only a few areas can be included as the real feature for distinguishing this category. In addition, the lighting conditions, background overlaps, articulation, and occluding accessories such as umbrellas and backpacks may cause changes to the silhouette of the pedestrian. Conventional pedestrian detection methods require complex feature extraction manually and have a limitation of slow processing time. However, modern convolutional neural network models such as Faster R-CNN [1], R-FCN [2], Mobilenets [3] and SSD [4] has been found to have more accurate and fast performances. But the appropriate tradeoff between speed and accuracy for each of the models has not been evaluated for the application of pedestrian detection. The detection speed is calculated in terms of seconds per frame (SPF) and the accuracy metric used is mean average precision (mAP).

The State-of-the-art pedestrian detection frameworks depend on region proposal algorithms to estimate the location of the pedestrians. After the advancement of networks like SPPnet [5] and Fast R-CNN [6] the running time was reduced thereby making the region proposal computation to impede. Th Region

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 365–379, 2018.
https://doi.org/10.1007/978-3-319-96136-1_29

Proposal Network (RPN) introduced cost-free region proposals that share convolutional features with the detection network. Thus, Faster R-CNN is composed of a deep fully convolutional network that proposes regions, and these regions are used by the Fast R-CNN detector to predicts object bounds and objectness scores at each position. In case of PASCAL VOC 2007, 2012, and MS COCO datasets this model achieved high detection accuracy with only 300 proposals per image. The region-based, fully convolutional networks (R-FCN), in contrast to the region-based detectors such as Fast/Faster R-CNN have all computations shared on the entire image. Meanwhile, the Faster R-CNN applies a costly per region network on the image large number of times thereby increasing the computational burden. The R-FCN model can also use the Residual Networks (ResNets) [7] as a fully convolutional image classifier backbone for various detection applications. To manage real-world applications such as self-driving car and augmented reality, the recognition tasks must be performed on a computationally limited platform. MobileNets are such small, low latency models that can be easily matched to the design requirements for mobile and embedded vision applications. They were primarily constructed by performing depth wise separable convolutions. It was subsequently used in Inception models [8] to reduce the computation in the first few layers. While accurate, the current approaches have been too computationally intensive for embedded systems and, too slow for real-time applications, even with high-end hardware. A single deep neural network named SSD was introduced which achieved 77.2% mAP for 300300 input and 79.8% mAP for 512512 input on VOC2007, outperforming a comparable state-of-the-art Faster R-CNN model. Several attempts were made to build faster detectors, but so far, significantly increased speed comes only at the cost of significantly decreased detection accuracy.

In this paper, the speed/accuracy trade-off of modern detection systems are explored in the application of pedestrian detection. The test time performances, duration of training, learning rate and loss functions for each model are compared to find the optimal detector for this application. Here lesser training time denotes faster convergence to a more accurate model using fewer parameters. This reduces the complexity of the system as well as prevents overfitting. The Faster R-CNN, R-FCN, and SSD at a high level consist of a single convolutional network and are trained with a mixed regression and classification objective thereby making it easier to compare and analyze these systems. The implementations of these architectures were done in TensorFlow which finally provided the tradeoff results for various detection systems.

2 Model Architecture

The detection framework consists of the convolutional object detectors with different meta architectures and feature extractors. The meta architectures include proposal based methods such as R-FCN and Faster R-CNN and proposal free method with SSDs. In Faster R-CNN the computational burden is shared and the region proposals are generated using neural networks. Hence this architecture has improved efficiency while in R-FCN the removal of fully-connected layers improves speed as well as accuracy. SSD uses different bounding boxes and small convolutional filters for prediction. It achieves high detection speed even while using relatively low-resolution input. In particular, several pre-trained models such ResNet, MobileNet, and Inception are used for feature extraction. Thus the proposed network uses a combination of these architectures and trained them with the multi-task loss function for object detection and localization.

2.1 Modern Convolutional Detectors

Single Shot Detector. SSD uses a single deep Neural Network to detect the objects. SSD makes the output space of bounding boxes discrete to form a set of default boxes. These default boxes are formed over different aspect ratios and scales for each of the feature map location. During the prediction, scores are generated for each of the object category in each default box and the box dimensions are adjusted to fit the object shape better [4].

Single Shot Multibox Detector is a feed forward Convolutional network producing a collection of bounding box which have fixed-size. The scores are generated for the presence of an object class instance in those boxes, following which a non-maximum suppression step is done which produce the final detection. The base network layers are based on standard architecture used for higher quality image classification. Feature Layers are added to the truncated base network which decrease in size progressively as shown in Fig. 1. Hence, the detections at multiple scales is possible. Using a set of Convolutional filters each of the added feature layers formulates a fixed set of detection predictions. For a feature Layer, the parameters used for prediction is mainly a kernel of size 3 × 3 × p (the feature layer having size m × n with p channels) which produces either a score for a category or the shape offset with respect to the default box position relative to feature map location [4]. Each feature map cell has a default bounding box assigned to it. For each of these feature map cell the offsets relative to the default box shapes in the cell are calculated. Also, the scores that indicate the presence of a class instance in these boxes are predicted for each class. At the training time, the default boxes are first compared with the ground truth boxes following the calculation of model loss. The model loss calculated is hence a weighted sum between Localization loss [6] which corresponds to the shape offset and confidence loss calculated from the confidences for all object categories.

SSD Training Objective. The training objective is obtained from the Multibox Objective [9,10]. The Objective Loss function is the weighted sum of Localization Loss (loc) and Confidence Loss (conf) [1] given by Eq. (1).

$$L(x,c,l,g) = \frac{1}{N}(L_{conf}(x,c) + \alpha L_{loc}(x,l,g)) \tag{1}$$

N is the number of matched default boxes. Localization Loss is computed in the form of a smooth L1 loss [6] between the parameters corresponding to the predicted box (l) and the ground truth box (g) [4]. Confidence loss is mainly the loss over different class confidences (c). For N = 0, loss is set to zero.

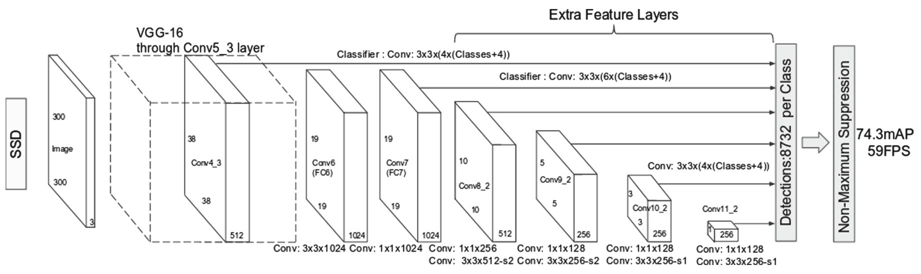


Fig. 1. Architecture of SSD

Faster R-CNN. Faster R-CNN (Faster region-based convolutional neural networks) is a single, unified network mainly used for object detection [1]. Fast R-CNN developed before the Faster R-CNNs were able to achieve near real-time rates using very deep networks [11], ignoring the time spent on region proposals. [1] The Region proposal step was computationally expensive with a huge running time.

Faster R-CNN has two modules. The first module being the deep fully convolutional network proposing regions. The second module is the detector module from Fast R-CNN [6]. These detectors use the proposed regions. The recently popular term attention [12] mechanisms, the Region Proposal Network tells the Fast R-CNN detector module where to look at [1]. Figure 2 shows the description of the layers. A Region Proposal Network (RPN) takes an image as input. The output is a set of rectangular proposals for the objects with a score depicting the objects presence. RPN is modelled using a fully connected convolutional network. The region proposals are generated by sliding a small network over the convolutional feature map. This feature map is obtained from the previous convolutional layer. The small network takes in an n × n spatial window of the feature map as the input. These windows are then mapped to a lower dimensional feature. The Low-dimensional features are then fed to two fully connected layers namely box regression layer (reg) and box-classification layer (cls). [1] In Fig. 3, for k locations corresponding to maximum possible proposals reg layers have 4k outputs showing coordinates of k boxes whereas cls layer has 2k scores which help in estimating the probability of object occurrence for each proposal. These k proposals are parameterized with respect to k reference box called as anchor. Anchor based approach helps in addressing the multiple scale and aspect ratio issues. This design hence avoids extra cost for addressing scales.

The loss function for an image while training RPN is defined as:

$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{2}$$

i is the index for an anchor in mini batch and p_i is the predicted probability of anchor I being an object. The ground-truth label p_i^* is 1 if the anchor is positive and 0 if negative. t_i represents the coordinated and t_i^* is the ground-truth box counterpart. L_{cls} is the log loss over two classes (whether object or not) and L_{reg} is the regression loss. The two terms obtained from the cls and reg are normalized by mini-batch size. During training either the RPN is trained first and the proposals are used to train the Fast R-CNN or the alternative technique involves both the models to be merged into one network during training which is shown in Fig. 2.

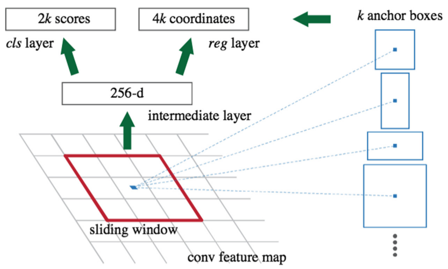


Fig. 2. Region Proposal Network (RPN)

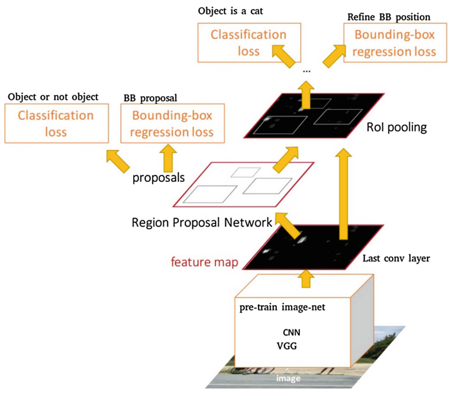


Fig. 3. Architecture of Faster R-CNN

R-FCN (Region Based Fully Convolutional Network). R-FCN (Region based Fully Convolutional Network) consists of shared, fully convolutional architectures. The translation variance is incorporated into FCN. To do this a set of position-sensitive score maps are constructed by using a bank of specialized convolutional layers as the FCN output. [2] The architecture mainly contains Region Proposal Network which extracts candidate regions. The Region Proposal Network used here are fully Convolutional. These features are then shared between the RPN and R-FCN (similar to [1]). The proposal regions constitute the RoIs, The R-FCN architecture classifies these objects into categories and background.

In the R-FCN all the weight layers are convolutional and takes in the entire image as shown in Fig. 4. R-FCN ends with a position-sensitive RoI pooling Layer [2]. The last convolution layer outputs are aggregated in this layer. This layer generates scores for each region proposal, hence making this layer to learn specialized position-sensitive score maps. The main difference of R-FCN from R-CNN is that all the layers are fully convolutional. The learning mechanism is similar.

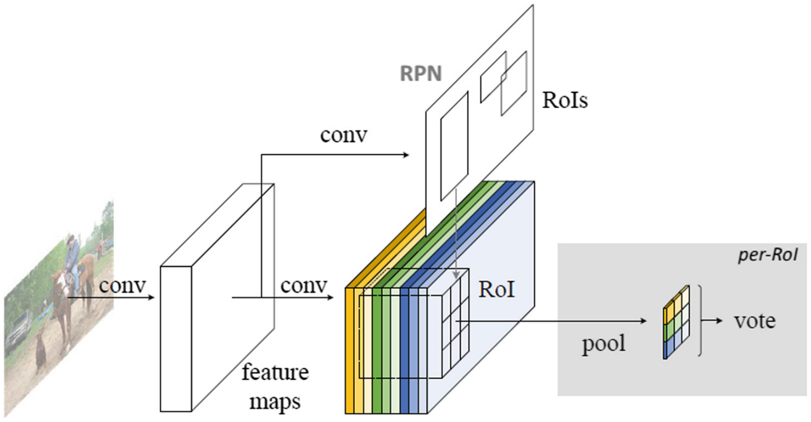


Fig. 4. Overall architecture of R-FCN

2.2 Feature Extractors

Mobilenet. MobileNet model uses depthwise separable convolutions. Depthwise separable convolutions factorizes a standard convolution into Depthwise Convolution and 1×1 convolution. The 1×1 convolutions are known as pointwise convolution [3]. The Depthwise convolution applies a single filter to each input channel. Pointwise convolution on the other hand applies 1×1 convolution where the outputs of depthwise convolution are combined. Pointwise convolution linearly combines the output from the depthwise Convolutional layer. Unlike basic convolution where the filters and inputs combine into a set of outputs in one step; in MobileNets the depthwise separable convolution splits this convolution process into two different layers. The first layer does filtering, and the second layer performs the combination. This mainly helps in reducing the computational cost and reduces the Model size. Both batchnorm and ReLU nonlinearities are used for both layers [3].

In the MobileNet structure, all the layers are built based on depthwise separable convolutional Layers except the first layer which is full convolution [3]. All Layers are followed by batchnormal and ReLU nonlinearity. The final Layer is the exception which has no nonlinearity but is followed by a softmax Layer instead which helps in classification [8]. MobileNet models were trained in TensorFlow [13] using RMSprop [14] with asynchronous gradient descent similar to Inception V3 [3,15].

Inception v2. This model mainly aims at utilizing added computation as efficiently as possible to perform factorized convolutions and aggressive regularization. The convolutional networks can be scaled up in an efficient way by using Inception concepts. Th convolutions involving a kernel greater than size 3×3 can be easily and efficiently performed using a series of smaller convolutions [15]. Further, factorization of convolutions and improved normalizations can be other tricks that are adopted to improve its efficiency.

Inception networks are fully convolutional, and each weight corresponds to multiplication per activation. If the factorization is done properly the parameters can be more disentangled and hence can lead to faster training. Inception-v2 network is 42 layers deep. The computational cost is only about 2.5 higher than that of GoogLeNet and is more efficient than VGGNet [15].

3 Experimental Setup

The pedestrian detector is trained on the Penn-Fudan dataset [16]. This database contains images of pedestrians are taken from scenes around campus and urban street. Each image will have at least one pedestrian in it. There is a total of 170 images in which 80% are used for training while the rest are used for testing. The convolutional neural network models are pre-trained on the COCO dataset, the Kitti dataset, and the Open Images dataset and hence lesser data is required for re-training the models. The detections on COCO dataset based on these models are shown in Fig. 5. In case of the COCO dataset, the different models provide a trade off between speed of execution and the accuracy in placing bounding boxes as shown in Table 1.

Further, in order to train the model, for each image, the width, height and each class with their bounding box parameters are required. Hence the input data is labeled manually using the graphical image annotation tool named LabelImg. It uses Qt for its graphical interface and the annotations for each image are saved as XML files in PASCAL VOC format. A labeled map associated with each of the datasets and this label map defines a mapping from string class name (person) to integer class ids which always start from id 1. During training, TensorFlow uses the TFRecord format in order to optimize the data feed. Initially, the XML files are converted to CSV files which are further converted to TFRecord files.

The pre-trained models such as SSD MobileNet, SSD Inception, Faster RCNN Inception, R-FCN Resnet101 and Faster RCNN Resnet101 are chosen for analyzing the dataset. Each model has corresponding checkpoint files as well as configuration files for the training process. The pre-trained model using the COCO dataset was designed to work in 90 categories. In the configuration files, transfer learning is done by removing the last 90 neuron classification layer of the network

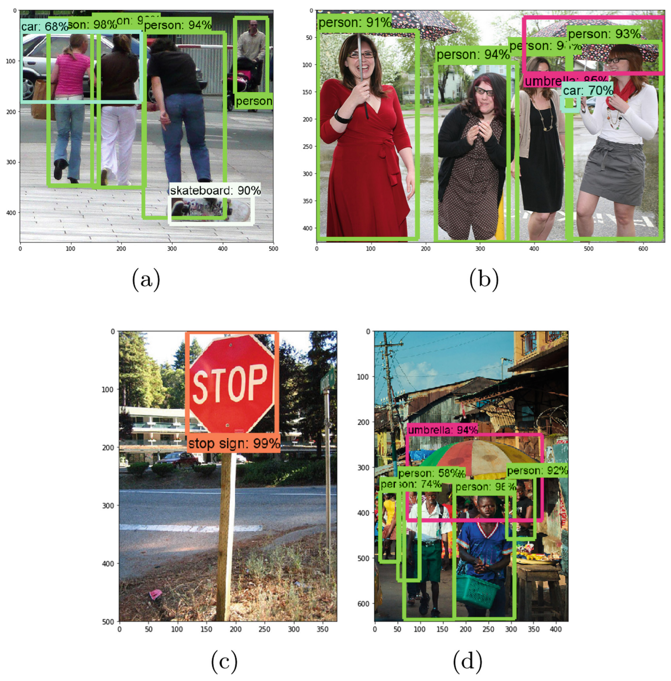


Fig. 5. Sample detection on COCO 2017 test images

Table 1. Performance comparison for COCO dataset

Model name	Speed	COCO mAP	Outputs
ssd_mobilenet_v1_coco	Fast	21	Boxes
ssd_inception_v2_coco	Fast	24	Boxes
faster_rcnn_inception_v2_coco	Medium	28	Boxes
rfcn_resnet101_coco	Medium	30	Boxes
faster_rcnn_resnet101_coco	Medium	32	Boxes

and replacing it with a new layer. Here this is implemented so that training will be quicker and the data required will be less. Thus the five pre-trained models are taken and the last layer is clipped off and replaced with the class of the current dataset. The SSD MobileNet and SSD Inception used a batch size of 10 while the rest of the models used a much lesser batch size of 3. Here each of the five models is trained for 5000 number of steps and these steps mainly depends on the size of the dataset. The initial learning rates are in the range of 0.002–0.004 for each of the models.

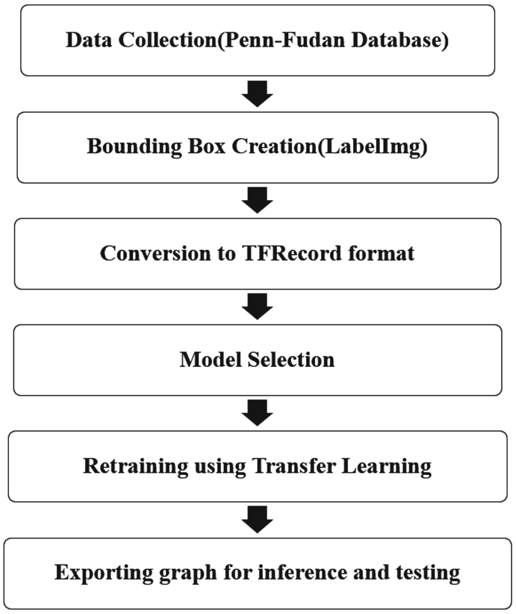

Data Collection(Penn-Fudan Database)
Bounding Box Creation(LabelImg)
Conversion to TFRecord format
Model Selection
Retraining using Transfer Learning
Exporting graph for inference and testing

Fig. 6. Experimental flow chart

The fine-tuning of an existing model is highly accurate and easy since most of the features that are learned by CNNs are often object agnostic. Thus all the feature detectors trained in the previous model are used to detect the new class. When the loss function for each of the models is around 1 or starts rising the training is stopped. Finally, the graph for inference is exported and the newly trained model is validated using the remaining 20% test data as well raw Google images. These images are in the RGB format with varying sizes and are in .png image format. Many of these images contain multiple pedestrians and those holding many occluding object artifacts such as bags and umbrellas. Each of the trained models is tested at multiple checkpoints to see which one performs the best. The entire experimentation is shown in the flowchart in Fig. 6.

4 Results

4.1 Detection Time on Test Images on CPU

The detection time per image was calculated for each model as shown in Table 2. It was seen that faster RCNN inception and SSD models are faster requiring around 2.5 s on average detection time per image. However faster RCNN resnet models has a higher computation burden and thereby require around 18 s.

The overall mAP calculation of different models is shown in Table 2. It is observed that faster rcnn inception outperformed other models with a higher accuracy value. Based on the ranking of detection scores for each class the performance of the detectors are evaluated using mean average precision over entire test data.

Table 2. Speed accuracy trade-off

Model name	Time (s)	mAP
ssd_mobilenet	1	0.78
ssd_inception_v2	1.2	0.92
faster_rcnn_inception	4.7	0.96
rfcn_resnet101	7.31	0.95
faster_rcnn_resnet101	18.23	0.95

4.2 Sample Detections

The visualization of detections in the test images of the Penn-Fudan dataset can be seen in Fig. 7. Thus a comparative analysis is done between the 5 models to find the optimum detector for the pedestrian detection application. The Penn-Fudan dataset consist of different categories of pedestrians such people carrying umbrellas, suitcases and other occluding objects. There also exist variable cases such as multiple people on the same screen and overlapping scenarios. Thus sample detections done on this dataset can be considered to be highly comparable to real life situations. From the figures it is clear that all the models perform consistently well except that SSD is unable to detect pedestrians which are farther away and it also shows low detection rate in case of overlaps. Further, sample detection using Faster R-CNN Inception on a Google image is shown in Fig. 10.

4.3 Total Loss Function

The loss function for the models is the total loss in doing detection and generating bounding box. Each of the models was trained for 5000 steps since the size of the dataset was comparatively small. The loss for SSD (both with ResNet 101 and Inception-v2) models started off with a value around 13 and converged around the value 2. In case of Faster R-CNN (both with ResNet 101 and Inception-v2) and R-FCN models, the loss function converged to a relatively smaller value around 0.2.

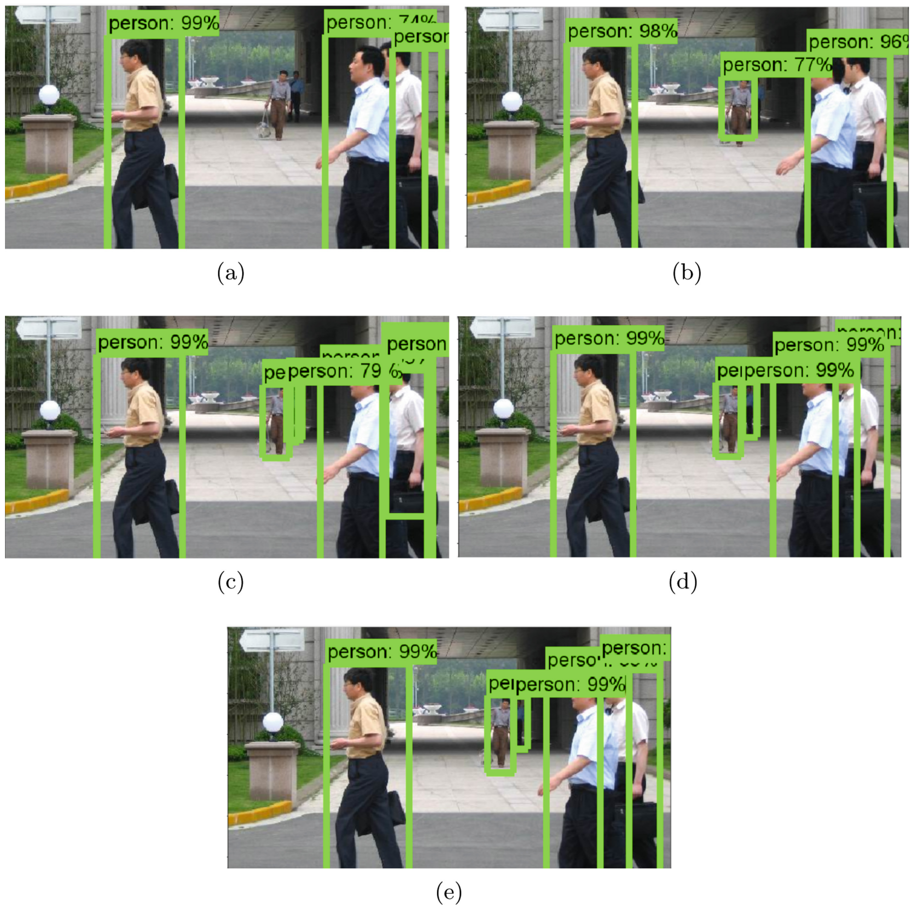


Fig. 7. Example detection for (a) SSD Mobilenets (b) SSD Inception (c) R-FCN Resnet (d) Fast RCNN Resnet (e) Fast RCNN Inception

Therefore, it can be seen that appropriate training was done for the latter models, till the loss function converged to a very less value, particularly below 1. The training loss corresponding to each of the models is shown in Fig. 8.

4.4 Learning Rate

Learning rate is a hyper-parameter used for adjustment of the weights in the network with respect to the loss gradient. Learning rate is a measure of travel down the slope to reach the local minima and it decides how quickly a model can converge to a local minima. From the above figures it is clear that better accuracy of Faster R-CNN Inception is due to its high learning rate compared

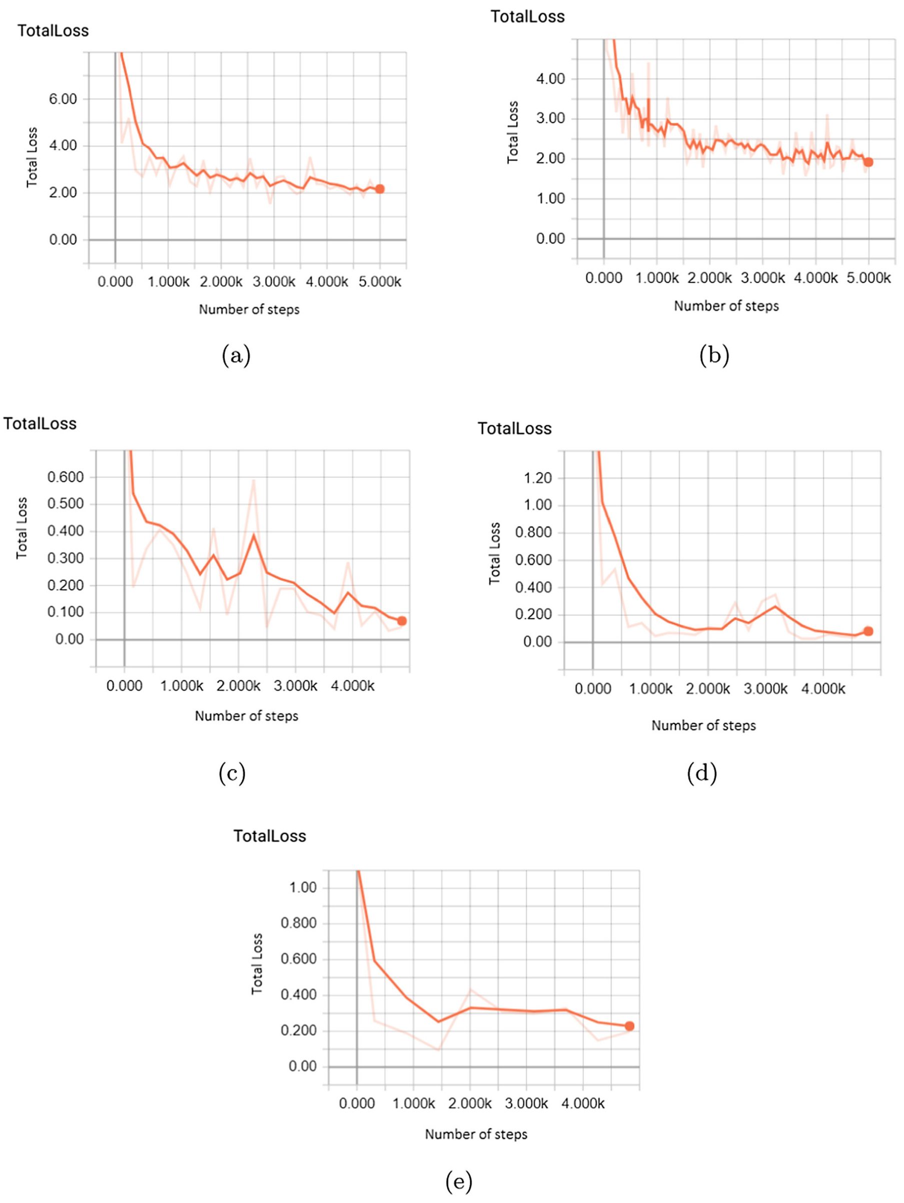


Fig. 8. Total loss for (a) SSD Mobilenets (b) SSD Inception (c) R-FCN Resnet (d) Fast RCNN Resnet (e) Fast RCNN Inception

to rest of the models. Lower value for learning rate means, the descent is slower along the slope. With the perfect learning rate a lesser training time can be achieved. The learning rates for different models are shown in Fig. 9.

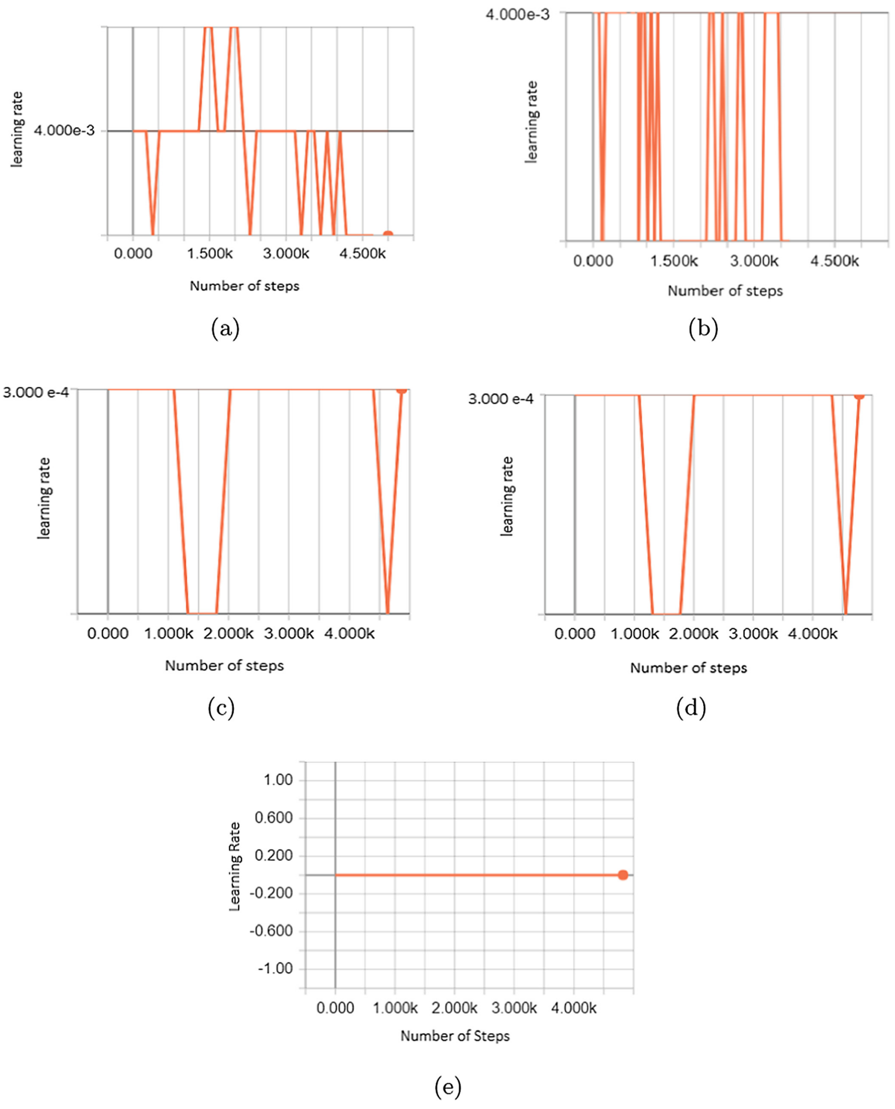


Fig. 9. Learning rates for (a) SSD Mobilenets (b) SSD Inception (c) R-FCN Resnet (d) Fast RCNN Resnet (e) Fast RCNN Inception

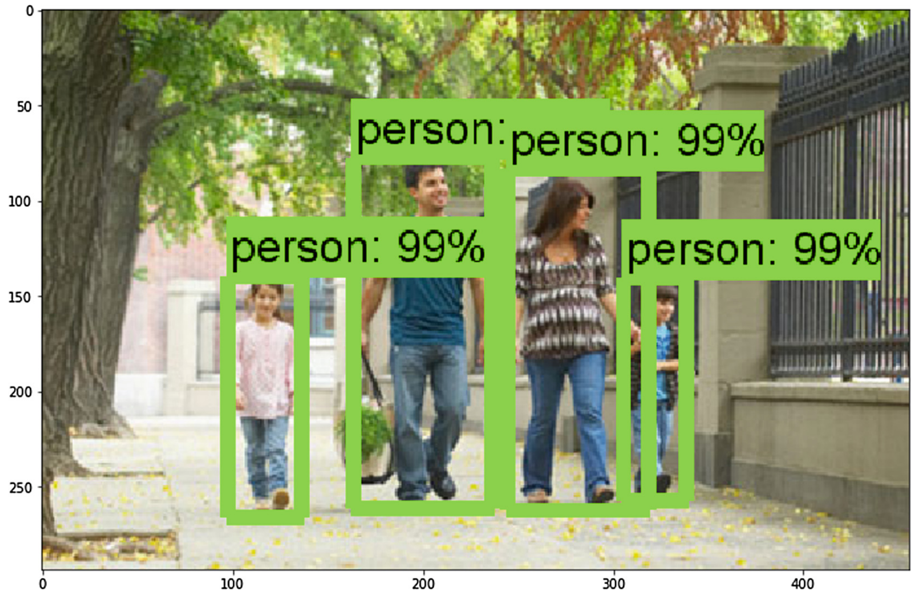


Fig. 10. Detection using Faster R-CNN on a Google Image

5 Conclusion

An experimental analysis was done on the modern object detection model to find the appropriate trade off in speed and accuracy. It was observed that Faster RCNN Inception model gave the most optimal values in speed and accuracy for the application of pedestrian detection. It was able to detect even far away people with overlaps with an accuracy of 96%. In addition, the rfcn resnet 101 and faster rcnn resnet 101 gave comparable accuracies but the test time was found to be very high. Thus the pedestrian can move out of the frame by the time the model tries to detect it. The SSD models were much faster to train but the detection rate was lesser compared to other models and occluded people were much difficult to detect using this model.

References

1. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
2. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing Systems, pp. 379–387 (2016)
3. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)

4. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: SSD: single shot MultiBox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
5. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8691, pp. 346–361. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10578-9_23
6. Girshick, R.: Fast R-CNN. arXiv preprint arXiv:1504.08083 (2015)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
8. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
9. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2147–2154 (2014)
10. Szegedy, C., Reed, S., Erhan, D., Anguelov, D., Ioffe, S.: Scalable, high-quality object detection. arXiv preprint arXiv:1412.1441 (2014)
11. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
12. Chorowski, J.K., Bahdanau, D., Serdyuk, D., Cho, K., Bengio, Y.: Attention-based models for speech recognition. In: Advances in Neural Information Processing Systems, pp. 577–585 (2015)
13. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., et al.: TensorFlow: large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)
14. Tieleman, T., Hinton, G.: Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA: Neural Netw. Mach. Learn. **4**(2), 26–31 (2012)
15. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
16. Wang, L., Shi, J., Song, G., Shen, I.: Object detection combining recognition and segmentation. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) ACCV 2007. LNCS, vol. 4843, pp. 189–199. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76386-4_17

Automated Machine Learning Algorithm Mining for Classification Problem

Meng-Sung Wu[1](✉) and Jun-Yi Lu[2]

[1] Industrial Technology Research Institute, Hsinchu, Taiwan
wums@itri.org.tw
[2] National Chiao Tung University, Hsinchu, Taiwan
alex42132000@hotmail.com

Abstract. Hyper-parameter optimization and the identification of the learning algorithm best suited to a particular dataset can be exceedingly difficult. Researchers have developed automated methods for the selection of an algorithm and the associated hyper-parameters; however, this approach is not necessarily applicable to other datasets. In this paper, we present a method for the selection of a learning algorithm while simultaneously setting the hyper-parameters in a two-stage process: (1) Identification of important hyper-parameters to streamline the optimization process, and (2) Heuristic formulation based on sequence analysis to limit the long-tuning time and identify the optimal algorithm/ hyper-parameter combination. The proposed method greatly reduces the training time without a significant loss of performance in classification tasks.

Keywords: Machine learning · Hyper-parameter · Algorithm selection Optimization

1 Introduction

Machine Learning (ML) plays an integral role in many aspects of businesses intelligence, recommendation systems, Fintech, and artificial intelligence. Research in ML has led to the development of numerous methods for the creation of classifiers [28]; however, it is not always clear which approach would be the most appropriate for a given dataset [29]. Previous work determined the best classifier for a variety of ML problems is compared to different techniques and over different dataset sizes [9]. Building an effective learning model, finding the right ML method, and fine-tuning the hyper-parameter settings is becoming increasingly important [15]. The tuning of hyper-parameters used to be based on human experience; however, identifying the optimal ML technique for a particular problem can be difficult even for experts, particularly as data grows in complexity. To address this gap, there have been big strides in the development of user-friendly machine learning software that can be used by non-experts [16, 27].

Hyper-parameter optimization strategies vary with regard to performance and cost. The conventional approach generally involves a grid search in order to enhance accuracy; however, the evaluation of all possible parameter combinations is also computationally very expensive. One alternative approach is to sampling different

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 380–392, 2018.
https://doi.org/10.1007/978-3-319-96136-1_30

parameter combinations use a random selection to cover the parameter space to a lesser extent [3]. Random search has been shown to match or exceed grid search while necessitating fewer function evaluations for some types of problem. The problem of hyper-parameter optimization has also been addressed using sequential model-based optimization (SMBO), Bayesian optimization methods [2, 26, 27] and the tree-structured Parzen estimator (TPE) [2]. These methods are meant to identify good configurations in less time than it would take using random search. Scikit-Learn [19] and Auto-WEKA [27] software enable the automatic configuration of the ML library to find the optimized combination of data preprocessing, hyper-parameter tuning, and model selection [2, 3, 14, 20, 26]. Nevertheless, differences in the size and characteristics of datasets influence the performance of optimization methods; i.e., no single method can achieve a high degree of efficiency and accuracy at the same time.

In this work, we present an automated method for the selection of the most appropriate ML algorithm as well as the optimization of its hyper-parameter settings specifically for a particular dataset. The configuration procedure first identifies important hyper-parameters in order to streamline the optimization process. We then implement an early stopping strategy [7] to progressively filter out unsuitable algorithms until only the best model remains [8, 24]. The main contributions of this paper are as follows: (a) the ordinary least square combine with Analysis of variance (ANOVA) is conducted for the influence of hyper-parameters so as to “hold on” only relevant parameters for optimization in each algorithm; (b) a heuristic based on sequence analysis is used to terminate long-tuning time procedures and find an optimal algorithm and its configuration hyper-parameter settings.

The remainder of this paper is organized as follows: In Sect. 2, we present an overview of existing methods used in the selection of machine learning algorithms and their associated hyper-parameters. Section 3 describes the methodology developed in this study. Section 4 presents a description of and the experiments used to evaluate the efficacy of the proposed scheme. Finally, Sect. 5 summarizes the paper and describes directions for future work.

2 Foundations and Related Work

2.1 Problem Statement

Classification is among the most important aspects of machine learning. Numerous learning algorithms have been developed for the task of classification, such as the Random Forest, SVM, decision trees, and neural networks [1]. As described in the Introduction, the characteristics of datasets largely determine which learning algorithm would be most appropriate for model training, and each learning algorithm requires optimization based on the selection of hyper-parameters. For example, the Random Forest algorithm proposed by Breiman [5] combines multiple tree predictors that depend on the value of the random vector sample, wherein majority voting is used to predict the results. Numerous hyper-parameters can be used to optimize the model, such as the depth of the trees, the number of trees in the forest, and the number of features to consider at each split. Even slight changes can greatly affect prediction accuracy [4]. In the past, the results were evaluated manually to evaluate the appropriateness of the learning algorithm and the effectiveness of the selected hyper-parameters. This approach is time-consuming and seldom provides acceptable results.

2.2 Algorithm Selection and Hyper-parameter Optimization Methods

Given a dataset $\mathcal{D}$ and a set of algorithms $\mathcal{A}$, the objective of the algorithm selection is to identify the algorithm $A^* \in \mathcal{A}$ is the best suited to the data. Let the data $\mathcal{D}$ split into a training set $\mathcal{D}_{train}^{(i)}$ and a validation set $\mathcal{D}_{valid}^{(i)} = D/\mathcal{D}_{train}^{(i)}$ for $i = 1, \cdots, k$. Using multi-fold cross validation [13], the algorithm selection problem can be written as follows:

$$A^* \in \operatorname{argmax}_{A \in \mathcal{A}} \frac{1}{k} \sum\nolimits_{i=0}^{k} \mathcal{S}\left(A, \mathcal{D}_{train}^{(i)}, \mathcal{D}_{valid}^{(i)}\right), \tag{1}$$

where $\mathcal{S}\left(A, \mathcal{D}_{train}^{(i)}, \mathcal{D}_{valid}^{(i)}\right)$ is the validation performance achieved by algorithm A when trained using $\mathcal{D}_{train}^{(i)}$ and evaluated on $\mathcal{D}_{valid}^{(i)}$.

Similarly, given an algorithm A and a dataset $\mathcal{D}$, suppose that $\boldsymbol{\Lambda} = \{\boldsymbol{\lambda}_1, \cdots, \boldsymbol{\lambda}_l\}$ denotes the hyper-parameters of an algorithm A, where each $\boldsymbol{\lambda}_m$ is a variable defining its respective domain. The goal in selecting hyper-parameter values can be formulated as follows:

$$\boldsymbol{\lambda}^* \in \operatorname{argmax}_{\boldsymbol{\lambda} \in \Lambda} \frac{1}{k} \sum\nolimits_{i=0}^{k} \mathcal{S}\left(A_{\boldsymbol{\lambda}}, \mathcal{D}_{train}^{(i)}, \mathcal{D}_{valid}^{(i)}\right), \tag{2}$$

where A_λ represents the algorithm A under a specific configuration of hyper-parameter values $\boldsymbol{\lambda}$. Then, given a set of algorithms with associated hyper-parameter space $\boldsymbol{\Lambda}^{(1)}, \cdots, \boldsymbol{\Lambda}^{(n)}$. The optimization problem involves the selection of an algorithm as well as the hyper-parameter values with the aim of achieving the best cross-validation performance:

$$A^*, \boldsymbol{\lambda}^* \in argmax_{\mathrm{A}^{(j)} \in \mathcal{A}, \boldsymbol{\lambda} \in \Lambda^{(j)}} \frac{1}{k} \sum\nolimits_{i=0}^{k} \mathcal{S}\left(\mathrm{A}_{\boldsymbol{\lambda}}^{(j)}, \mathcal{D}_{train}^{(i)}, \mathcal{D}_{valid}^{(i)}\right). \tag{3}$$

2.3 Hyper-parameter Search

The most common approaches to hyper-parameter optimization are grid search and random search [3]. Grid search involves the inspection of all possible permutations associated with the grid points; however, this scheme often performs poorly and does not scale to higher dimensions. To enhance efficiency, random search makes selections from a random distribution of alternatives in order to avoid having to conduct an exhaustive search of all possible hyper-parameters. Most existing automated hyper-parameter optimization methods use sequential model-based optimization (SMBO) methods, including Bayesian optimization [2, 26, 27] and the Tree-structured

Parzen Estimator (TPE) [2]. Bayesian Optimization techniques have been shown to outperform other state-of-the-art global optimization methods through the use of posterior distribution to calculate the latent variable and thereby sample the hyper-parameters used by the following step. To approximate the unknown objective function, Gaussian process (GP) have often been used as the probability model in the Bayesian optimization [26]. In this approach, a promising hyper-parameter configuration is suggested to test next by acquisition function. A popular acquisition function is the expected improvement (EI), defined as [12]

$$EI(x) = \mathbb{E}[\max(0, f(x) - f(\hat{x}))] = \int_{-\infty}^{\infty} \max(0, f(x) - f(\hat{x}))p(f(x)|x). \tag{4}$$

where $\hat{x}$ is the current optimal set of hyperparameters. TPE is a nonparametric approach to density estimation which explores intelligently the search space while narrowing down to the estimated best parameters. The TPE approach models the posterior $p(f(x)|x)$ by two separated models, non-parametric densities $p(x|f(x))$ and the configuration prior $p(x)$. A well-known implementation of TPE is Hyperopt library [14].

3 Methodology

In this work, we propose a sequential process by which to optimize the effects of machine learning. The proposed algorithm includes two phases, first determines important hyper-parameters by ordinary least square (OLS) to streamline the optimization process, and then a heuristic based on sequence analysis is used to terminate long-running training procedures and find an optimal algorithm and its configuration hyper-parameter settings. Here, the OLS method [23, 25] was adopted to analyze the coefficient between the various hyper-parameters and the overall accuracy. As a form of multiple linear regression, this method is highly effective in evaluating relevance in the multi-factorial analysis [18, 22]. The loss function of OLS involves minimizing the sum of squares in order to obtain an estimate of the unknown variables in the linear model. We also established an automatic early stopping scheme inspired by developments in neural networks [6] to prevent over-fitting in the gradient descent process when the accuracy of the results fails to increase within a given number of epochs [21].

3.1 Identification of Essential Hyper-parameters

The goal of identifying important hyper-parameters is to exclude hyper-parameters that are less relevant to the objective and to select the most influential hyper-parameters to construct the classifier. In general, there is no efficient way of direct optimum hyper-parameter identification. Hence we usually rely on some heuristics to overcome the complexity of exhaustive search. Ordinary least squares (OLS) [23] is one of the commonly used heuristic methods for estimating the unknown parameters in a linear regression model. The general equation of OLS model can be written as

$$Y = X \cdot \delta + \varepsilon, \tag{5}$$

where Y is a vector representing the dependent variable, δ is the coefficient vector and X is a matrix of regressors.

Analysis of variance (ANOVA) [11] can be performed to check whether the regression model is statistically significant or not (F-test and corresponding p-value). We adopted it for the influence of hyper-parameters so as to "hold on" only relevant parameters for optimization in each algorithm. The F-test of hypothesis and the coefficient of determination (R-squared) are used to evaluate the effect of the model. If the p-value of the F-test is lower than the significance level α and R-squared value is high, then indicates that there is a strong relationship between performance and hyper-parameters [10]. Under the above conditions, T-test for each hyper-parameter estimates can be performed to see whether they are statistically significant or not. If the p-value of T-test is below significance level β, then that particular hyper-parameter is shown to have a positive effect on the performance of the learning model. This hyper-parameter is retained and the others (that fail to meet the level of significance) are removed. We examine numerous algorithms in the first process and record the highest performance achieved by each of them. If the highest performance achieved by a particular algorithm falls below the average performance of all algorithms, then that particular algorithm is removed in order to reduce computational complexity in the following step.

3.2 Algorithm Evaluation and Selection

Early Stopping Method. The optimization phase explores the hyperparameter space and finds candidate configurations that return a high performance. However, there may be a large number of candidate configurations in the optimization process, it is necessary to design an effective discarded or promote configuration policies. In this work, we present a solution that implements early stopping for hyper-parameter search techniques and we show that this effectively reduces the effect of computation time. Given an algorithm A with associated hyper-parameter configuration $\mathbf{\Lambda} = \{\boldsymbol{\lambda}_1, \cdots, \boldsymbol{\lambda}_l\}$, we decompose the hyperparameter configuration into g-epochs, $\mathbf{\Lambda} = \left\{e^{(1)}, \cdots, e^{(g)}\right\}$. Suppose that $e^{(u)} = \left\{\boldsymbol{\lambda}_{(u-1)\cdot h+1}, \cdots, \boldsymbol{\lambda}_{u\cdot h}\right\}$, for $u = 1, \cdots, g$, where each epoch represents a combinations of h hyper-parameter configuration. Here, we calculated the running best performance value of the epoch. If the optimal value has found during a specified number of epochs is worse than the best value of all the previously completed tests, this algorithm interrupts the optimized process.

Recursive Filtering for Candidate Algorithms. To find good configurations faster, we support parallel runs for algorithm selection on a single machine. Masini and Bientinesi [17] present efficient mechanisms that allow running more than one algorithm in parallel. The well-known optimization tools [14, 27] provide efficient parallelization work for performing hyperparameter optimization and model selection. In this paper, the chosen strategy analyzes the local optima found by recursive filtering and selecting the best algorithms for the removal of the non-essential algorithms. We evaluate the performance of each algorithm and compare it after running m-times to further narrow down the range of selected algorithms and passing them to the next run level. That procedure is repeated on the pruned set until one fully-optimized algorithm remained. The process described above is presented in Algorithm 1.

Algorithm 1: Recursive algorithm selection procedure

Given : machine learning algorithm set $\mathcal{A} = \{A^{(1)}, \cdots, A^{(n)}\}$ with associated hyper-parameter space $\Lambda^{(1)}, \cdots, \Lambda^{(n)}$, where $\Lambda^{(i)} = \{\lambda_{i,1}, \cdots, \lambda_{i,l}\}$ for $i = 1, \cdots, n$, training set $\mathcal{D}_{train}$ and validation set $\mathcal{D}_{valid}$

```
B ← φ; C ← φ; t = 1;
Repeat
    for every $A^{(i)} \in \mathcal{A}$ do in parallel
        for $j \leftarrow ((t-1) \cdot m + 1)$ to $(t \cdot m)$ do
            $S_{i,j}$ = Eval $(A^{(i)}, \lambda_{i,j}, \mathcal{D}_{train}, \mathcal{D}_{valid})$
        end

    // find the largest element in each algorithm
        $A_{\lambda}^{(i)} \leftarrow A_{\lambda}^{(i)} | \mathcal{S}_{A^{(i)},\lambda} = \max\left(\mathcal{S}\left(A^{(i)}, \lambda_{(t-1)\cdot m+1}\right), \cdots, \mathcal{S}\left(A^{(i)}, \lambda_{t\cdot m}\right)\right)$
    end

    if $C = \phi$ then
        for every $A_{\lambda^*}^{(i)}$ do
            $A_{\lambda^*}^{(i)} \leftarrow A_{\lambda}^{(i)}, \mathcal{S}_{A^{(i)},\lambda^*} \leftarrow \mathcal{S}_{A^{(i)},\lambda}$
            $C \leftarrow C \cup \{A_{\lambda^*}^{(i)}, \mathcal{S}_{A^{(i)},\lambda^*})\}$
        end
    else if $\mathcal{S}_{A^{(i)},\lambda} > \mathcal{S}_{A^{(i)},\lambda^*}$
            update $A_{\lambda^*}^{(i)} \leftarrow A_{\lambda}^{(i)}, \mathcal{S}_{A^{(i)},\lambda^*} \leftarrow \mathcal{S}_{A^{(i)},\lambda}$
        end
    end
    $B \leftarrow C$

    // filtering
    for every $A^{(i)} \in C$ do
        if $\mathcal{S}_{A^{(i)},\lambda^*} <$ AVG $(\mathcal{S}_{A,\lambda^*})$
            $C \leftarrow C \setminus A^{(i)}, \mathcal{A} \leftarrow \mathcal{A} \setminus A^{(i)}$
        end
    end
    $t = t + 1$

Until $\mathcal{A} = \phi$
Return $B$
```

4 Experimental Evaluation

4.1 Corpora and Setup

In the experiments, we conducted the evaluation of proposed methods for classification task using three standard collections from the UCI repository [9]: the Iris dataset, Handwritten Digit dataset and the Mushroom dataset. The Iris dataset consists of 50 samples/instance from each of three species that totals to 150 records. The Handwritten Digit dataset include 8×8 image of integer pixels in the range 0…16, we'd have to transform it into a feature vector with length 64. The Mushroom dataset is composed of records of different types of mushrooms, the goal is to determine a mushroom is poisonous or edible. The dataset was randomly divided into two parts, in which 70% was used as training data and 30% was used as variation test data. Table 1 summarizes the number of classes and the number of discrete and continuous attributes found in the datasets.

Table 1. Datasets from the UCI repository.

Name	Data type	Attributes	Class	Training	Test
Iris	Real	4	3	100	50
Handwritten digit (HD)	Integer	64	10	1198	599
Mushroom	Binary	22	2	5416	2708

In this study, popular machine learning algorithms were considered for evaluating their performance in terms of classification error: Decision Tree (DT), Random Forest (RF), Gradient Boost Decision Tree (GBDT), Passive Aggressive Regression (PAR), stochastic gradient descent (SGD) and Support Vector Machine Classifier (SVC) [27]. All experiments are performed using ten-fold cross-validation over training and variation set. We selected four different hyper-parameters optimization methods as the foundation of the experiment: Grid Search, Random Search (RS), Bayesian Optimization (BO) and Tree of Parzen Estimators (TPE). Several parameters need to be determined in the experiments. The significance level was set as $\alpha = \beta = 0.01$.

4.2 Experimental Results

First of all, we ran ordinary least squares method with functional ANOVA approach to identify the most important hyperparameters of the algorithm for each dataset, and list these in Table 2. For each algorithm, we report the number of original hyper-parameters and the number of not influential hyper-parameters filtered out in each dataset. Overall, in this experiment, we observed that there is at least a hyper-parameters are filtered out, as shown in bold. For illustration, Table 3 shows the F-test of hypothesis and the coefficient of determination (R-squared) in more detail which using GBDT algorithm for Iris dataset. The R-squared value indicates that the hyper-parameter model can explain more than 94% of the variation in performance. The overall model achieved two stars for the significant level, which means that it can be used to remove non-essential hyper-parameters. The *p*-value of max_depth and learning_rate is smaller than 0.01, which means that this two hyper-parameter have considerable influence on performance and are therefore retained while the others are discarded.

Table 2. Comparison of the number of influential hyperparameters for different algorithms in each dataset.

Algorithm	Num. of original hyper-parameter	Iris	HD	Mushroom
Decision tree	4	**2**	**2**	**3**
Random forest	4	**3**	**2**	**3**
GBDT	4	**2**	4	**2**
SGD	2	**1**	2	2
SVC	3	**2**	3	3
PAR	2	2	1	2

Table 3. The result of the OLS analysis using GBDT algorithm for Iris dataset.

	R-squared	Adj. R-squared	F-statistic	Prob. (F-statistic)
Entire	0.955	0.948	138	4.15E−17
	Coef	**Std err**	**t**	**P > |t|**
max_depth	0.0314	0.005	5.777	0
max_features	0.0603	0.024	2.491	0.019
min_samples_split	0.0079	0.005	1.484	0.15
learning_rate	34.9795	9.352	3.74	0.001

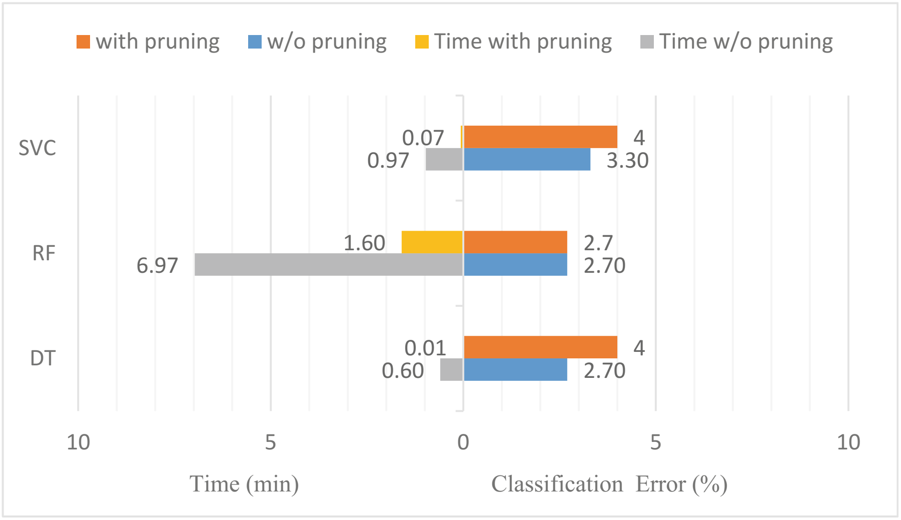


Fig. 1. Performance of with/without pruning for different algorithm evaluations with the grid-based hyper-parameter for the Iris dataset, the right side axis is classification error and the left side axis is search time (minutes).

Next, based on the results in Table 2, we illustrate the impact of search time and classification errors on DT, SVC, GBDT and RF combined grid search in Iris datasets. As the results shown in Fig. 1, the performance is slightly reduced but greatly improves the hyper-parameter tuning process. This trend is the same for all four classifiers but is most clear with GBDT (not shown in the figure). For GBDT, search time dropped from 88.43 min to 2.914 min (an improvement of 92.3%), while the classification error increased only by 0.6 (2.7% → 3.3%). Similarity, we could see a roughly 4–13× speedup in execution time for other algorithms as shown in Fig. 1.

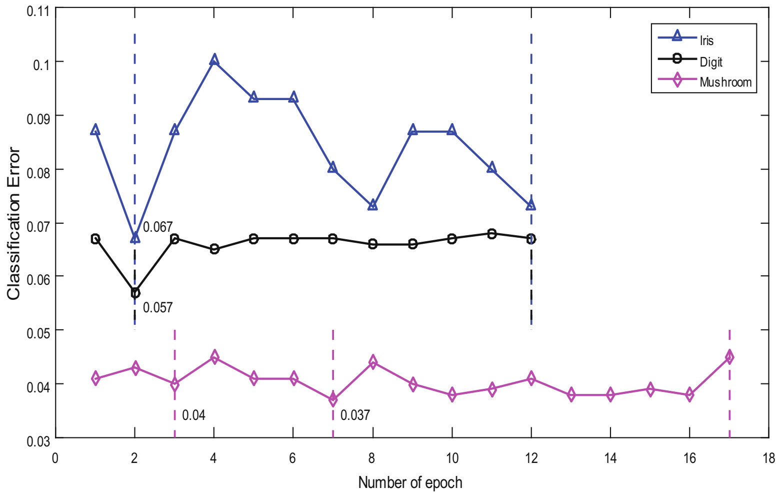


Fig. 2. Comparison of the performance in each epoch by using the GBDT method with a random search for three datasets). The left side axis represents the classification error and the dashed line indicates the classification error of the epoch exceeds the highest value obtained in previous epochs.

In the previous evaluation, the improvements of each algorithm on all dataset are obtained due to the identification of influence of hyper-parameter in the search space. Hereafter, we implement early stopping concept for all methods, if the performance does not improve on the validation set during ten epochs, the process stops. Each epoch represents a combination of h hyper-parameter configuration setting. $h = 10$ during the random search and $h = 2$ during the process using BO and TPE. The maximum number of epochs is limited to 50. We illustrate the use of GBDT algorithm with random search optimization method in three datasets, as shown in Fig. 2. We found that Iris and Mushroom dataset stopped at the 12th epoch, while the Digit dataset stopped at the 17th epoch, as shown in Fig. 2. To evaluate the effectiveness of the early stopping (ES)-based heuristic, we measured 200 hyper-parameter configurations as a baseline. Results of the early stopping experiments are presented in Fig. 3. Each dataset was run through each different hyper-parameters optimization method with a various ML algorithm. Figure 3 revealing that different model perform very differently on different datasets. In Fig. 3, we note a mean 48.9% decrease in total epochs across these three datasets and note that validation error only gets slightly worse. We can see that Bayesian Optimization (BO) converges to a good result much faster than Tree of Parzen Estimators (TPE) and random search. With this early stopping strategy, the BO achieves 73.1% reduction in tuning times for the Mushroom dataset. The relative improvements by BO over RS and TPE are 34.2% and 46.6%, respectively. In summary, the classifiers to be the bests are Support Vector Machine (SVC) and Gradient Boost Decision Tree (GBDT). On average, these methods achieve in average 2.8% of the classification error. Followed by random forest (RF), achieves more than 3.7% of the classification error. For the Mushroom dataset, the classification error is 0.3% for GBDT and 1.1% for SVC, respectively. Similarly, the classification error by GBDT and SVC are 1.3% and 1.2% for the Handwritten Digit dataset, respectively.

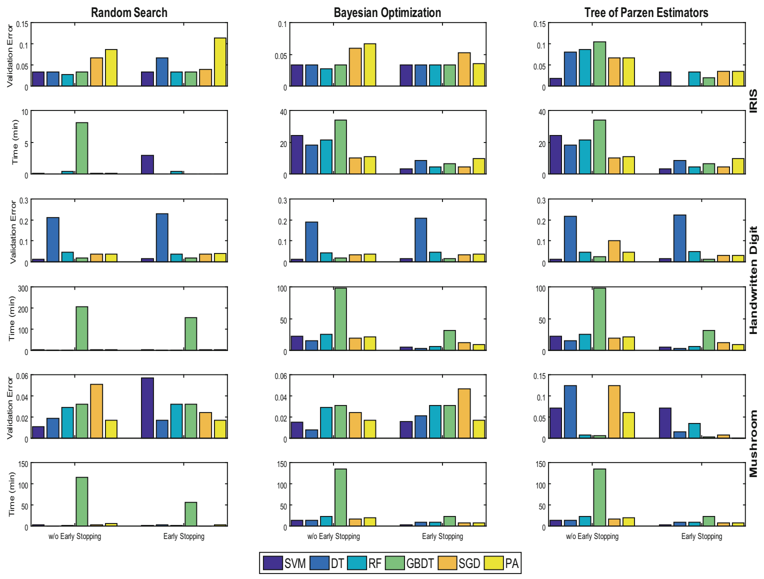


Fig. 3. The effect of early stopping strategy. ML algorithms were compared across several datasets with various hyper-parameter optimization methods. Classification error on a validation dataset is shown for each combination.

At last, we compare the effects of parallel strategies on various optimization methods for the choice of algorithms as shown in Table 4. From Table 4, we can see that TPE converges faster than random search and BO optimization. For Mushroom dataset, we compare 62.99 min to 5.57 min, an 11× speedup. By contrast, a 2× speedup in the case of random search and a 5× speedup in the case of BO. The conventional TPE search with no early stopping takes 103.29 min. In this case, the best classification performance is obtained by using the RF classifier, which achieves 0.2% of the classification error. In some way, the same improvements have been made in

Handwritten Digit (HD) and Iris datasets. For HD dataset, the SVC is the best classifier achieve a classification error rate 1.28%. When applied to the Iris dataset, obtains an classification error rate of 2.3% by using the GBDT.

Table 4. Performance (search time) of the three hyper-parameter tuning methods on each datasets using early stopping without/with the parallelize strategies for algorithm selection.

Dataset	RS		BO		TPE	
	ES	Parallel+ES	ES	Parallel+ES	ES	Parallel+ES
Iris	5.08	2.1	43.05	8.09	43.73	3.76
HD	215.21	93.32	78.90	35.4	129.55	28.98
Mushroom	113.58	54.62	101.80	22.17	62.99	5.57

5 Conclusion and Future Work

The paper presents a framework for selecting a suitable learning algorithm with corresponding algorithmic parameters for a given set of data. Two-process optimization method was proposed for algorithm selection and hyper-parameter tuning to efficiently. We first apply configuration procedures that determine important hyper-parameters and showed the importance of effects through a function OLS and ANOVA methods. Then, we use a heuristic based on sequential analysis to early-stop long-tuning procedures and support parallel runs to filter out unappropriated algorithms until ultimately the best model remains. In the light of this, we can drop up some methods which have lower accuracy and irremediable to attain the same effect as the best one to reduce the unnecessary waste of resource. In our future work, we intend to extend this study by more optimization algorithms. Furthermore, we plan to find the best setting for hyper-parameters within the large-scale dataset.

Acknowledgments. We are very grateful to the anonymous reviewers and editor. This work was sponsored by Ministry of Economic Affairs, Taiwan, R.O.C. through project No. G301ARY910 conducted by ITRI. The author would like to thank the anonymous reviewers for their detailed comments and suggestions that helped to improve the paper.

References

1. Ali, S., Smith, K.: On learning algorithm selection for classification. Appl. Soft Comput. **6**, 119–138 (2006)
2. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: Advances in Neural Information Processing Systems, pp. 2546–2554 (2011)
3. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. J. Mach. Learn. Res. **13**, 281–305 (2012)
4. Bernard, S., Heutte, L., Adam, S.: Influence of hyperparameters on random forest accuracy. In: Benediktsson, J.A., Kittler, J., Roli, F. (eds.) MCS 2009. LNCS, vol. 5519, pp. 171–180. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02326-2_18

5. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001)
6. Caruana, R., Lawrence, S., Giles, L.: Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping. In: Proceedings of the 13th International Conference on Neural Information Processing Systems, pp. 381–387 (2000)
7. Collobert, R., Bengio, S.: Links between perceptrons, MLPs and SVMs. In: Proceedings of the Twenty-First International Conference on Machine Learning, pp. 23–30 (2004)
8. Duvenaud, D., Maclaurin, D., Adams, R.: Early stopping as nonparametric variational inference. In: Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, pp. 1070–1077 (2016)
9. Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems? J. Mach. Learn. Res. **15**, 3133–3181 (2014)
10. Fidler, F., Thompson, B.: Computing correct confidence intervals for ANOVA fixed- and random-effects effect sizes. Educ. Psychol. Measur. **61**, 575–604 (2001)
11. Hooker, G.: Generalized functional ANOVA diagnostics for high-dimensional functions of dependent variables. J. Comput. Graph. Stat. **16**, 709–732 (2007)
12. Jones, D., Schonlau, M., Welch, W.: Efficient global optimization of expensive black box functions. J. Glob. Optim. **13**, 455–492 (1998)
13. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 1137–1143 (1995)
14. Komer, B., Bergstra, J., Eliasmith, C.: Hyperopt-Sklearn: automatic hyperparameter configuration for scikit-learn. In: ICML Workshop on AutoML (2014)
15. Lin, S., Ying, K., Chen, S., Lee, Z.: Particle swarm optimization for parameter determination and feature selection of support vector machines. Expert Syst. Appl. **35**, 1817–1824 (2008)
16. Luo, G.: A review of automatic selection methods for machine learning algorithms and hyper-parameter values. Netw. Model. Anal. Health Inform. Bioinform. **5**, 18 (2016)
17. Masini, S., Bientinesi, P.: High-performance parallel computations using python as high-level language. In: Guarracino, Mario R., et al. (eds.) Euro-Par 2010. LNCS, vol. 6586, pp. 541–548. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21878-1_66
18. McElroy, F.: A necessary and sufficient condition that ordinary least-squares estimators be best linear unbiased. J. Am. Stat. Assoc. **62**, 1302 (1967)
19. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, É.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
20. Pedregosa, F.: Hyperparameter optimization with approximate gradient. In: Proceedings of the International Conference on Machine Learning, pp. 737–746 (2016)
21. Prechelt, L.: Automatic early stopping using cross validation: quantifying the criteria. Neural Netw. **11**, 761–767 (1998)
22. Puntanen, S., Styan, G.: The equality of the ordinary least squares estimator and the best linear unbiased estimator. Am. Stat. **43**, 153 (1989)
23. Rao, C.: Linear Statistical Inference and Its Applications. Wiley, New York (2002)
24. Schreuder, M., Höhne, J., Blankertz, B., Haufe, S., Dickhaus, T., Tangermann, M.: Optimizing event-related potential based brain–computer interfaces: a systematic evaluation of dynamic stopping methods. J. Neural Eng. **10**, 036025 (2013)
25. Skipper, S., Josef, P.: Statsmodels: econometric and statistical modeling with python. In: Proceedings of the 9th Python in Science Conference, pp. 57–61 (2010)

26. Snoek, J., Larochelle, H., Adams, R.: Practical Bayesian optimization of machine learning algorithms. In: Proceedings of the 25th International Conference on Neural Information Processing Systems, pp. 2951–2959 (2012)
27. Thornton, C., Hutter, F., Hoos, H., Leyton-Brown, K.: Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 847–855 (2013)
28. Witten, I., Frank, E., Hall, M.: Data Mining: Practical Machine Learning Tools and Techniques. Elsevier, Amsterdam (2011)
29. Wolpert, D.: The lack of a priori distinctions between learning algorithms. Neural Comput. **8**, 1341–1390 (1996)

Enhancing Outlier Detection by an Outlier Indicator

Xiaqiong Li[1], Xiaochun Wang[1(✉)], and Xia Li Wang[2]

[1] School of Software Engineering, Xi'an Jiaotong University, Xi'an 710049, China
xiaqiongli@stu.xjtu.edu.cn, xiaocchunwang@mail.xjtu.edu.cn

[2] School of Information Engineering, Changan Univeristy, Xi'an 710061, China
xlwang@chd.edu.cn

Abstract. Outlier detection is an important task in data mining and has high practical value in numerous applications such as astronomical observation, text detection, fraud detection and so on. At present, a large number of popular outlier detection algorithms are available, including distribution-based, distance-based, density-based, and clustering-based approaches and so on. However, traditional outlier detection algorithms face some challenges. For one example, most distance-based and density-based outlier detection methods are based on k-nearest neighbors and therefore, are very sensitive to the value of k. For another example, some methods can only detect global outliers, but fail to detect local outliers. Last but not the least, most outlier detection algorithms do not accurately distinguish between boundary points and outliers. To partially solve these problems, in this paper, we propose to augment some boundary indicators to classical outlier detection algorithms. Experiments performed on both synthetic and real data sets demonstrate the efficacy of enhanced outlier detection algorithms.

Keywords: Outlier detection · Distance-based outlier detection
Density-based outlier detection · Boundary detection · *k*-Nearest neighbors

1 Introduction

With the rapid development of information technology, a large amount of information has been produced from the real word. How to find import and useful information from these massive and multi-dimensional data has become an urgent problem. Therefore, data mining and database technologies come into being consequently.

In practice, data often come from different information individuals, departments, enterprises, and countries. These complex data sets may contain a small portion of data which differ significantly from other data objects in behavior or model. These data objects are called outliers. A general intuition of what constitutes an outlier was given by Hawkins in 1980. "Outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism" [1].

In many fields, outliers are more important than normal data, as they imply some useful information.

At present, the study on outlier detection becomes very active. Many outlier detection algorithms have been proposed. Outlier detection methods can be divided into distribution-based methods, depth-based methods, distance-based methods, density-based methods and clustering-based etc. However, none of them have been proved to be completely applicable to all the situations. Each type of outlier detection algorithms has its advantages and disadvantages. In distribution-based methods, an object is considered as an outlier if it deviates too much from a standard distribution (e.g., normal, Poisson, etc.) [2]. However the underlying distribution is usually unknown and there are many practical applications which do not follow a standard distribution. As a result, distribution-based methods have a limited number of applications. Being an improvement, depth-based methods assign a depth value to each data object and map it to the corresponding layer in the two-dimensional space. Data objects in the shallow layers are more likely to be outliers than those in the deeper ones. Unfortunately, these methods suffer high computational complexity for data of more than three dimensions. Distance-based methods, also known as adjacency-based methods, believe that data objects are outliers if they are far away from the majority of data points and address more globally-oriented outliers in databases [3]. However, distance-based methods are often accompanied by the problem that the values of k have a great influence on the results. Density-based methods usually assign to each data object a measure of outlier degree as the classic LOF algorithm does and then regard those data objects which possess largest outlier degrees as outliers [4]. In comparison to distance-based methods, these methods address more locally-oriented outliers. Finally, clustering-based methods obtain outliers as a by-product and regard those data items that reside in the smallest clusters as outliers [5].

However, traditional outlier detection algorithms face some challenges. For one example, most distance-based and density-based outlier detection methods are based on k-nearest neighbors and therefore, are very sensitive to the value of k. For another example, some methods can only detect global outliers, but fail to detect local outliers. Last but not the least, in many existing outlier detection algorithms, the boundary points are mistakenly classified to be outliers. To partially solve these problems, in this paper, we propose to augment classical outlier detection algorithms with some boundary indicators so as to enhance traditional methods for outlier detection. When compared with some classical outlier detection algorithms on sample datasets, the enhanced method is more accurate with less sensitivity to k.

The rest of the paper is organized as follows. In Sect. 2, we review some existing work on classic outlier detection algorithms. We then present our proposed enhancer in Sect. 3. In Sect. 4, a performance evaluation is conducted and the results are analyzed. Finally, conclusions are made in Sect. 5.

2 Related Work

If outliers exist in a data set, they will stay far away from other data points. Thinking about outliers in this way, Knorr and Ng proposed distance-based outlier detection method in 1998. Given a distance measure defined on a feature space, "an object O in a dataset T is a DB(p, D)-outlier if at least a fraction p of the objects in T lies greater than distance D from O", where the term DB(p, D)-outlier is a shorthand notation for a Distance-Based outlier (DB-outlier) detected using parameters p and D [3]. There are two classical algorithms based on this concept. "Given two integers, n and k, a Distance-Based outlier is the data item whose average distance to their k-nearest neighbors is among top n largest ones [6]" (referred to as "DB") and "Given two integers, n and k, a Distance-Based outlier is the data item whose distance to their k-th nearest neighbor is among top n largest ones [7]" (referred to as "DB-Max").

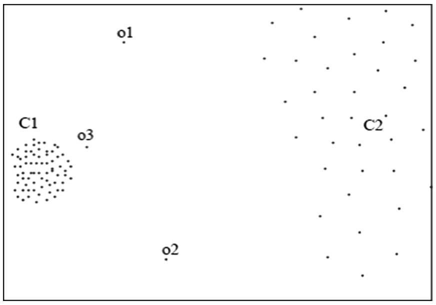


Fig. 1. A classic example of a local outlier.

The realization of distance-based outlier detection method is simple, but it is difficult to solve outlier detection problem in datasets with complex densities, as illustrated in Fig. 1. To overcome this limitation, in 2000, Breunig et al. proposed the density-based outlier detection method by introducing an outlier factor for each data item, called Local Outlier Factor (LOF), which is a ratio between the local density of an object and the average of the local densities of its k nearest neighbors [4]. It represents the degree of separation of an object relative to its local area. The top n objects are returned as outliers because the higher value of a data object's LOF means the higher possibility of its being an outlier.

In 2006, Wen Jin et al. presented a new density-based outlier detection algorithm named INFLO to solve the problem when outliers exist in the location where the density distributions in the neighborhood are significantly different [8]. This method considers the union of a point's k-nearest neighbors and its reverse nearest neighbors to obtain a measure of outlierness. The reverse nearest neighborhood of a data point p is defined to consist of those of its k-nearest neighbors for which p is also among its k nearest neighbors.

In 2011, Huang et al. proposed a new approach for outlier detection, named RBDA [9]. RBDA method is based on a ranking measure that focuses on the question of whether a point is central from its nearest neighbors. The problem with RBDA is its high computation cost.

3 The Proposed Enhancer for Outlier Mining

3.1 A Simple Idea

Distance-based outlier detection methods are good at identifying global outliers. To identify the relatively small number of outliers, kNN for each data point is first computed, together with the corresponding distances between each data to their k nearest neighbors. The average of these distances or the k-th distance is used as an outlier score in distance based outlier detection method. However, there is no reason to assume that this must be the case for some outliers due to the existence of boundary data points. To face this challenge, an outlier indicator can be an aid. A problem with distance-based outlier detection algorithms is that these methods do not take the outlying degrees of a data point's k-nearest neighbors into consideration in the detection process. As a result, false positives can happen. For example, for the sample dataset shown in Fig. 2, though DB or DB-Max outlier scores can be calculated for each data point and four boundary data points of cluster C_2 can be mistakenly detected as DB or DB-MAX outliers, there are no outstanding outliers. To prevent the false positives from happening, there must be some ways to differentiate between boundary points and outliers existing in a dataset in the first place. To do so, as a first degree approximation, the distances of each data point and its kNN to their first nearest neighbor within a cluster can be assumed to follow a uniform distribution and the corresponding mean and standard deviation can thus be calculated. The ratio of the standard deviation over the mean can be used to judge to some degree whether outliers exist or not. For the sample dataset shown in Fig. 2, if k is set to be 2 (i.e., 2NN), the distance of data point o_1 to its first nearest neighbor is the same as that of data point o_2 to its first nearest neighbor and that of data point o_3 to its first nearest neighbor. The corresponding ratio of the standard deviation over the mean is 0, indicating there are not outstanding global outliers.

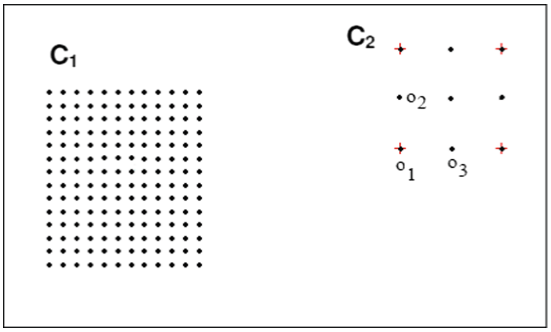


Fig. 2. An illustration of outlier indicator.

In the related work section, outlier definitions and some classical outlier mining algorithms are presented. These methods are able to identify some outliers, either global or local. However, for distance-based outlier detection methods, boundary points could be misclassified as outliers, while, for density-based outlier detection methods, global outliers may have low outlier scores and therefore be missed. Taking the dataset shown in Fig. 3 as an example, data point A is farthest away from its six closest neighbors and therefore should be identified as a global outlier. However, for k = 6, the LOF-based outlier detection method assigns a higher outlier score to data point B than to A and therefore, fails to identify A as the most significant global outlier. If global outlier detection is separated from local outlier detection, this will not happen.

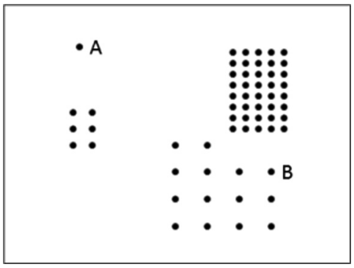


Fig. 3. An illustration of a difference between global outlier and local outlier definitions.

Based on these observations, we propose a new outlier detection algorithm, which enhances current state-of-the-art outlier mining methods by filtering out boundary points from outlier candidates, and formularize it in the following subsections.

3.2 Some Definitions

In this paper, we propose an outlier indicator to separate boundary points from being mixed with outliers to some extent.

Definition 1 (k-Distance of an object p). For any positive integer k, the k-Distance of object p, denoted as k-Distance(p), is defined as the *distance*(p, o), or simply, $d(p, o)$, between p and an object $o \in D$ such that:

(1) for at least k objects $o' \in D \backslash \{p\}$, $d(p, o') \leq d(p, o)$;
(2) for at most $k - 1$ objects $o' \in D \backslash \{p\}$, $d(p, o') < \mathrm{d}(p, o)$.

Definition 2 (k-Nearest Neighbors of an object p). For any positive integer k, given k-Distance(p), k-nearest neighbors of p contain the first k closest objects whose distance from p is not greater than k-Distance(p), denoted as $kNN_{k-Distance(p)}(p)$, for which, $kNN(p)$ is used as shorthand.

For outlier detection, we are more interested in those data points whose distance to its first nearest neighbor is significantly larger than the average value of the distances of the point's kNN to their first nearest neighbor. To quantify the significance of a data point's positioning outside some cluster, the uniform distribution is used as a first degree approximation for the distances associated with a data point and its kNN's to their first nearest neighbors. To distinguish between boundary points and outliers, we therefore formulate an outlier indicator using the distances associated with the first nearest neighbor of the data point and its kNN as in the following to focus our attention on the small number of outstanding global and local outliers.

Definition 3 (Outlier indicator of an object p). Given *k* nearest neighbors of an object p, in the following, *dist*[*0*] denotes the distance of an object p to its nearest neighbor, and *dist*[*i*] denotes the distance of its i-th nearest neighbor to its corresponding nearest neighbor, the proposed outlier indicator of an object p, $\mathrm{SOM}_{\mathrm{nn-dist}}(\mathrm{p})$, is defined based on these distances in the following,

$$Mean_{nn-dist}(p) = \frac{1}{k+1}\sum_{i=0}^{k} dist[i] \tag{1}$$

$$Std_{nn-dist}(p) = \sqrt{\frac{1}{k+1}\sum_{i=0}^{k} \left(dist[i] - Mean_{nn-dist}(p)\right)^2} \tag{2}$$

$$SOM_{nn-dist}(p) = \frac{Std_{nn-dist}(p)}{Mean_{nn-dist}(p)} \tag{3}$$

To manifest the effectiveness of the outlier indicator, for the test sample dataset shown in Fig. 4, we calculate the outlier indicators for all the data points and use the sum of their mean and standard deviation as a threshold to highlight the potential outliers. The results shown in Fig. 4 demonstrate that outliers O_1 and O_2, denoted by red color, are well identified because their indicator is much larger than the others.

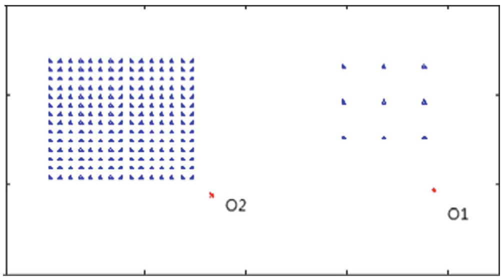


Fig. 4. An illustration of the effect of outlier indicator. (Color figure online)

3.3 Our Proposed Outlier Detection Algorithm

To find global outliers, we follow the notion of kNN based distance outlier definition and calculate the distance-based outlier factors (i.e., DB-MAX) for all the data points, sort them in a non-increasing order, and searching for the largest factor values. If their corresponding outlier indicators are significantly larger than a threshold value, SOM, top n data points can be regarded as the global outliers. For local outliers, we follow the notion of kNN based density-based outlier definition and calculate the LOF outlier factors for all the data points, sort them in a non-increasing order, and searching for the largest factor values. We combine three proposed factors to create our kNN-based outlier detection algorithm. To improve the readability, our proposed outlier detection algorithm is presented in a pseudo code format in Table 1.

Table 1. A combined outlier detection algorithm

Input:	*S*: a set of *N* data objects;
	k: the number of nearest neighbors;
	SOM: threshold
	n: the required number of top outliers.
Output:	*Index*: the indices of top-*n* outliers
Begin:	
1:	Compute *k* nearest neighbors for each data point;
2:	Compute the global and local outlier scores, DB-MAX and LOF, and outlier indicators;
3:	Compute the *mean* and *std* of the outlier indicators, and the threshold value, SOM;
4:	Sort the outlier scores, DB-MAX and LOF, in a non-increasing order;
5:	While(*Index*.size<n)
6:	{
7:	if(DB-MAX.next>SOM) *Index*.push_back(DB-MAX.next.index);
8:	if(LOF.next>SOM) *Index*.push_back(LOF.next.index);
9:	}
10:	Return *Index*.
End	

To determine the threshold of indicators in a data set, let the outlier indicators of all points in dataset be computed, based on which the average of the indicators, mean, and the corresponding standard deviation, std, are calculated. The threshold of indicators, SOM, for finding potential outliers is defined as,

$$SOM = mean(\text{indicators}) + f \times std(\text{indicators}) \tag{4}$$

To summarize, the numerical parameters the algorithm needs from the user include the data set, S, the loosely estimated number of outliers (i.e., the percentage of outlier candidates in the original data set), n, and the number of nearest neighbors, k.

4 Experiments and Results

In this section, we compare the effectiveness of the proposed outlier detection method with several state-of-the-art outlier detection methods, including the DB method, the DB-max, the LOF, the INFLO and RBDA methods, on several different datasets. In the first set of experiments, two 2-dimensional synthetic data sets are used to show that our proposed outlier detection method augmented with outlier indicator can outperform classical outlier detection algorithms in classification accuracy. Further, it is important for an outlier detection method to work effectively on real-world data sets. Therefore, in the second set of experiments, a real high-dimensional data sets obtained from the UCI Machine Learning Repository [10] are used to check the effectiveness of this study and to illustrate the effectiveness of our method in real-world situation. All the data sets are briefly summarized in Table 2. We implement all the algorithms in java and perform all the experiments on a computer with AMD A6-4400M Processor 2.70 GHz CPU and 4.00G RAM. The operating system running on this computer is Windows 7. In our evaluation, we focus on the outlier detection accuracy rate of these outlier detection algorithms on different data sets. The results show that, overall, our proposed outlier detection algorithm is superior over other state-of-the-art outlier detection algorithms.

Table 2. Description of all datasets

Data name	Data size	Dimension	#of outliers
syn_Data1	82	2	10
syn_Data2	473	2	6
LYMPHOGRAPHY	148	18	6

4.1 Performance of Our Algorithm on Synthetic Data

In this subsection, we use two synthetic datasets to show that the proposed outlier detection method performs better than traditional outlier detection methods. The two synthetic datasets, syn_Data1 and syn_Data2, are shown in the first plot of Figs. 5 and 6, respectively. For this set of experiments, the parameter *k*'s is set to be 3 for all the methods and the results for syn_Data1 and syn_Data2 are plotted in Figs. 5 and 6, respectively.

The first synthetic dataset, syn_Data1, consists of 82 instances, including six single outliers (i.e., A, B, C, D, E and F), and four clusters of different densities with 36, 8, 12 and 16 uniformly distributed instances. From the results depicted in Fig. 5, we can see that DB and DB-Max have the same ranks for A, D, E and F, but can not mine the two local outliers, that is, B and C. RBDA, INFLO, LOF and our outlier detection method detect all six outliers correctly. The plot at bottom left corner shows the corresponding $SOM_{nn-dist}$ values (which are actually 0 for boundary and inner points) thresholded by Eq. (4), which correctly identifies the six outliers.

The second synthetic dataset, syn_Data2, consists of 473 instances, including six outliers and five clusters of different densities clusters. A particular challenging feature of this data set is that three denser clusters are buried into one sparse cluster on the

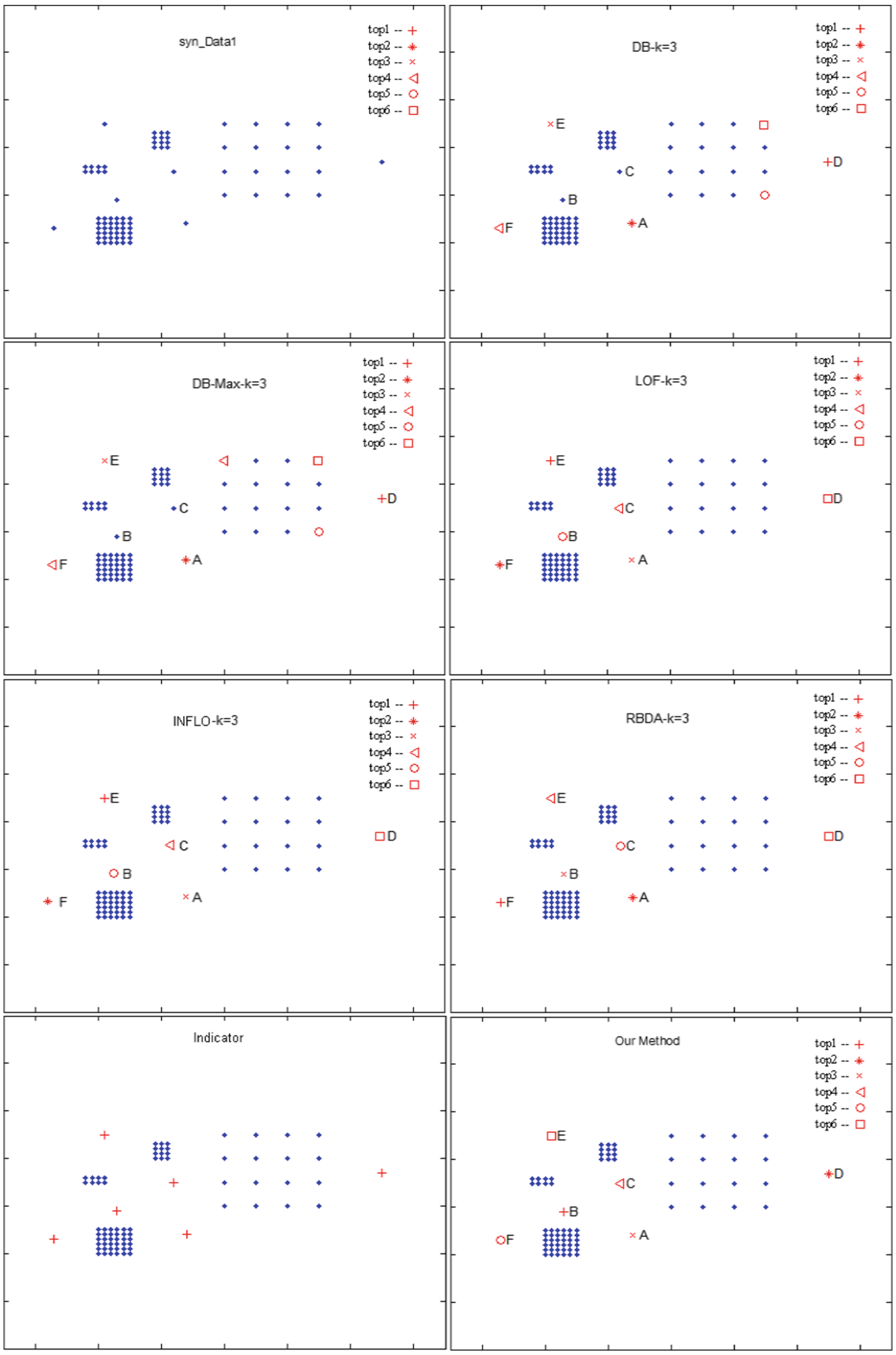


Fig. 5. The outlier detecting results on syn_Data1 for $k = 3$.

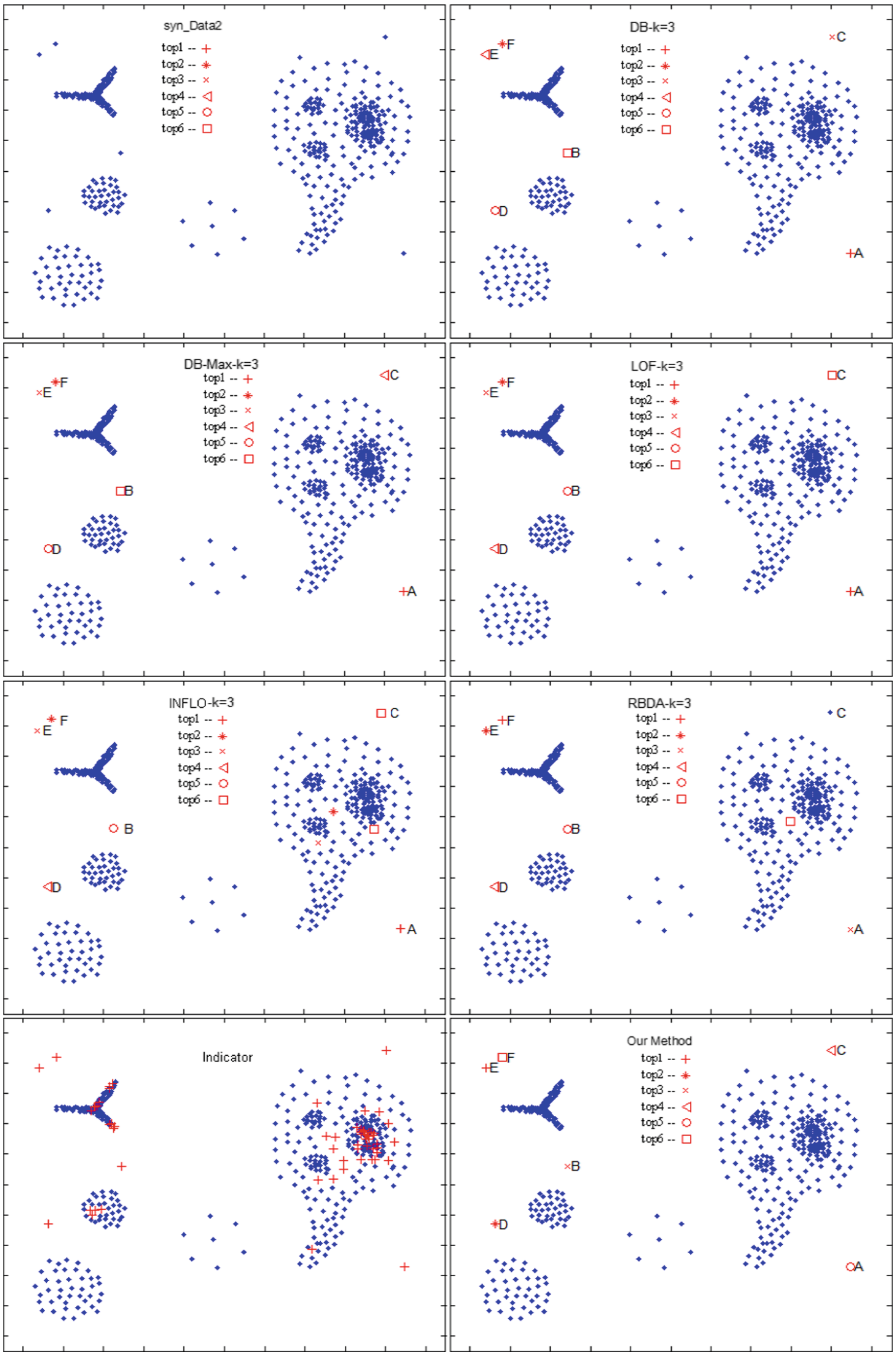


Fig. 6. The outlier detecting results on syn_Data2 for $k = 3$

upper right corner. From the results depicted in Fig. 6, we can see that this is a global outlier detection situation while the detection process is disturbed by the immediate connection of clusters with different densities. For detecting top 6 outliers, RBDA misses C but all other methods, that is, DB, DB-Max, LOF, INFLO and our method detect all six outliers correctly but with different rankings. The plot at bottom left corner shows the corresponding $SOM_{nn-dist}$ values thresholded by Eq. (4), which correctly identifies the six outliers.

To summarize, it can be observed from Figs. 5 and 6 that our method has no problems detecting all outliers and clearly offers the best ranking in three synthetic datasets while all other methods do not perform competently with detecting all the outliers one way or the other. The advantage of our outlier detection factors is very evident on these 2-dimensional data sets.

4.2 Performance of Our Algorithm on Real Data

It has been pointed out by Aggarwal and Yu that one way to test how well an outlier detection algorithm works is to run the method on the dataset and test the percentage of points which belongs to the rare classes [11]. In order to test how well our outlier indicator works on real dataset, we compare its ability in finding outliers with other methods in a real dataset, LYMPHOGRAPHY, which is downloaded from UCI [10]. This dataset has 148 instances with 18 attributes and contains a total of 4 classes. Classes 2 and 3 have 81 and 61 instances, respectively. The remaining two classes have totally 6 instances (2 and 4, respectively) and are regarded as outliers (i.e., rare classes) for they are small in size.

To quantitatively measure the performance of an outlier detection method, a popular metric, called recall, is used. Assuming that a dataset $D = D_o \cup D_n$ where D_o denotes the set of all outliers and D_n denotes the set of all normal data. Given any integer $m \geq 1$, if O_m denotes the set of outliers among objects in the top *m* positions returned by an outlier detection scheme, recall is defined as,

$$recall = \frac{|O_m|}{|D_o|} \tag{5}$$

In Eq. (5), recall shows the percentage of detected outliers in all outliers.

Table 3 shows the experimental results of the proposed outlier detection method in comparison with five other methods, DB, DBMax, LOF, INFlO, RBDA respectively, for four values of k's (i.e., 7, 10, 20, 30) and six values of m's (6, 7, 8, 9, 10, 15). In the table, n denotes the correct number of outliers among returned m ones, and r denotes the corresponding recall. From the experimental results, it can be seen that the proposed method mines all the outliers correctly for all m's and all k's and thus performs the best. RBDA method and LOF method perform next since they mine outliers as well as our method for k = 20 and k = 30 but does not do well in cases for k = 7 and k = 10. Overall, with increasing k's, RBDA, LOF and INFLO methods work better and better while DB and DB-Max work worse and worse.

Table 3. Experimental results for LYMPHOGRAPHY data

m	DB		DB-Max		LOF		INFLO		RBDA		OUR	
	n	r	n	r	n	r	n	r	n	r	n	r
k = 7												
6	5	0.83	5	0.83	5	0.83	4	0.67	5	0.83	**6**	**1.00**
7	**6**	**1.00**	5	0.83	5	0.83	5	0.83	5	0.83	**6**	**1.00**
8	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	5	0.83	**6**	**1.00**	**6**	**1.00**
9	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
10	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
15	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
k = 10												
6	5	0.83	5	0.83	5	0.83	4	0.67	5	0.83	**6**	**1.00**
7	**6**	**1.00**	5	0.83	5	0.83	5	0.83	**6**	**1.00**	**6**	**1.00**
8	**6**	**1.00**	5	0.83	**6**	**1.00**	5	0.83	**6**	**1.00**	**6**	**1.00**
9	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
10	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
15	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
k = 20												
6	5	0.83	5	0.83	**6**	**1.00**	5	0.83	**6**	**1.00**	**6**	**1.00**
7	5	0.83	5	0.83	**6**	**1.00**	5	0.83	**6**	**1.00**	**6**	**1.00**
8	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
9	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
10	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
15	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
k = 30												
6	5	0.83	5	0.83	**6**	**1.00**	5	0.83	**6**	**1.00**	**6**	**1.00**
7	5	0.83	5	0.83	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
8	5	0.83	5	0.83	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
9	**6**	**1.00**	5	0.83	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
10	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**
15	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**	**6**	**1.00**

5 Conclusions

Traditional distance based and density based outlier detection algorithms can effectively detect two different kinds of outliers separately but not both at the same time.

Further, classical distance based outlier detection algorithms do not differentiate global outliers from boundary data points. To partially circumvent these problems, in this paper, we have proposed a novel outlier detection approach which can detect both global and local outliers in a separate and simultaneous way and, when augmented with an outlier indicator, can outperform traditional outlier detection approaches. To demonstrate the utility of our proposed outlier detection mechanism, a detailed comparison is performed with state-of-the-art distance-based and density-based outlier detection methods. Experimental results show that our algorithm is able to rank the best candidates for being an outlier with high recall.

Acknowledgment. The authors would like to thank the Chinese National Science Foundation for its valuable support of this work under award 61473220 and all the anonymous reviewers for their valuable comments.

References

1. Hawkins, D.M.: Identification of Outliers. Monographs on Applied Probability and Statistics. Chapman and Hall, London (1980)
2. Barnett, V., Lewis, T.: Outliers in Statistical Data, vol. 3. Wiley, New York (1994)
3. Knorr, E.M., Ng, R.T.: Algorithms for mining distance-based outliers in large datasets. In: Proceedings of the 24th VLDB Conference, New York, USA, pp. 392–403 (1998)
4. Breuning, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 93–104 (2000)
5. Jiang, M.F., Tseng, S.S., Su, C.M.: Two-phase clustering process for outliers detection. Pattern Recogn. Lett. **22**, 691–700 (2001)
6. Angiulli, F., Pizzuti, C.: Fast outlier detection in high dimensional spaces. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) PKDD 2002. LNCS, vol. 2431, pp. 15–27. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45681-3_2
7. Ramaswamy, S., Rastogi, R., Shim, K.: Efficient algorithms for mining outliers from large data sets. In: Proceedings of the ACM SIGMOD Conference, pp. 427–438 (2000)
8. Jin, W., Tung, A.K.H., Han, J., Wang, W.: Ranking outliers using symmetric neighborhood relationship. In: Ng, W.-K., Kitsuregawa, M., Li, J., Chang, K. (eds.) PAKDD 2006. LNCS (LNAI), vol. 3918, pp. 577–593. Springer, Heidelberg (2006). https://doi.org/10.1007/11731139_68
9. Huang, H., Mehrotra, K., Mohan, C.K.: Rank-based outlier detection. J. Stat. Comput. Simul. **83**(3), 1–14 (2013)
10. UCI: The UCI KDD Archive, University of California, Irvine, CA. http://kdd.ics.uci.edu/
11. Aggarwal, C., Yu, P.: Outlier detection for high-dimensional data. In: Proceedings of the 2001 ACM SIGMOD Conference (SIGMOD 2001), Santa Barbara, CA, USA, pp. 37–46 (2001)

A Semi-supervised Approach to Discover Bivariate Causality in Large Biological Data

Nataliya Sokolovska[1](✉), Olga Permiakova[2], Sofia K. Forslund[3,4,5,6,7], and Jean-Daniel Zucker[8]

[1] Sorbonne University (Paris 6, UPMC), INSERM, Paris, France
nataliya.sokolovska@upmc.fr
[2] University Grenoble Alpes, CEA, BIG/BGE/EDyP, Grenoble, France
[3] Experimental and Clinical Research Center, a Cooperation of Charité-Universitätsmedizin Berlin and Max Delbruck Center for Molecular Medicine, 13125 Berlin, Germany
[4] Max Delbruck Center for Molecular Medicine in the Helmholtz Association, 13125 Berlin, Germany
[5] Charité-Universitätsmedizin Berlin, Corporate Member of Freie Universität Berlin, Humboldt-Universität zu Berlin, and Berlin Institute of Health, 10117 Berlin, Germany
[6] Berlin Institute of Health (BIH), Berlin, Germany
[7] European Molecular Biology Laboratory, Genome Biology Unit, 69117 Heidelberg, Germany
[8] IRD, INSERM, Bondy, Paris, France

Abstract. An important question in microbiology is whether treatment causes changes in gut flora, and whether it also affects metabolism. The reconstruction of causal relations purely from non-temporal observational data is challenging. We address the problem of causal inference in a bivariate case, where the joint distribution of two variables is observed. The state-of-the-art causality inference methods for continuous data suffer from high computational complexity. Some modern approaches are not suitable for categorical data, and others need to estimate and fix multiple hyper-parameters.

In this contribution, we focus on data on discrete domains, and we introduce a novel method of causality discovering which is based on the widely used assumption that if X causes Y, then $P(X)$ and $P(Y|X)$ are independent. We propose to explore a semi-supervised approach where $P(Y|X)$ and $P(X)$ are estimated from labeled and unlabeled data respectively, whereas the marginal probability is estimated potentially from much more (cheap unlabeled) data than the conditional distribution. We validate the proposed method on the standard cause-effect pairs. We illustrate by experiments on several benchmarks of biological network reconstruction that the proposed approach is very competitive in terms of computational time and accuracy compared to the state-of-the-art methods. Finally, we apply the proposed method to an original medical task where we study whether drugs confound human metagenome.

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 406–420, 2018.
https://doi.org/10.1007/978-3-319-96136-1_32

1 Introduction

Inferring causal directions between two variables from observational biological data in absence of time series or controlled perturbation is a challenging problem. In the past decades, the attention to the problem of discovering causality has grown due to necessity to reveal causality in real life applications. In particular, in the medical domain, revealing causal relations from a data set can help to improve clinical diagnostics, and to increase the quality of treatment and medication.

Mechanisms of action of many prescribed drugs remain unclair. Metformin is the most prescribed treatment for the type 2 diabetic patients, since it is relatively cheap, safe, and its important beneficial effects on blood glucose and cardiovascular parameters have been shown [27]. Nowadays, the main hypothesis of metformin action is that the drug mediates its antihyperglicemic effects by suppressing hepatic glucose output via the activation of AMP-activated protein kinase (AMPK)-dependent and AMPK-independent pathways in the liver [12]. However, recently some studies [14] confirmed hypotheses that metformin also acts through pathways in the gut.

In this paper, our goals are:

- to develop a robust causality inference method, since biological data are always limited and noisy,
- suspecting microbial mediation of therapeutic effects of metformin, test this hypothesis on a real data set.

Instead of learning causal structure of an entire dataset, some scientists focus on analysis of causal relations between two variables only. Modern conditional independence-based causal discovery methods (see, e.g., [17,30] for general overview) construct Markov equivalent graphs, and these methods fail in the case of two variables, since $X \to Y$ and $Y \to X$ are Markov equivalent.

In this contribution, we focus on a family of causality inference methods which are based on a postulate telling that if $X \to Y$, then the marginal distribution $P(X)$ and the conditional distribution $P(Y|X)$ are independent [8,9,26]. These approaches provide causal directions based on the estimated conditional and marginal distributions from observed non-temporal data. One of the most important problems in causality inference in a bivariate case, is to estimate the conditional and the marginal probabilities from noisy limited observed data as accurate as possible.

Deep learning methods [5] are becoming the preferred approach for various applications in artificial intelligence and machine learning, since they usually achieve the best accuracy. We are interested in particular in stochastic neural networks, whose activation units have a probabilistic element. Such a choice is motivated by the fact that conditional and marginal probabilities $P(Y|X)$ and $P(X)$ can be estimated by a deep model. In our experiments, we use Deep restricted Boltzmann machines (DRBM), originally introduced by [22], which is a deep stochastic model with one layer of visible units and several hidden units.

Our contribution is multifold:

– We introduce a novel semi-supervised method of inferring causal directions that allows to discover causal relations between different pairs of factors.
– We propose to estimate the conditional and marginal probabilities which are the key elements to infer directions, using the deep RBM.
– We illustrate by our experiments on benchmark data that the proposed method is computationally efficient and its performance is highly competitive compared to the state-of-the-art methods. It outperforms the existing methods in terms of accuracy.
– We consider a real biomedical problem of revealing causality in rich original metagenomic data. The interest to infer causality in metagenomic data is to verify hypotheses that drugs effects on metabolism are microbially mediated. We show that the proposed approach is efficient on the real complex data, and discuss the obtained results.

The paper is organized as follows. Related work discusses the state-of-art methods of the bivariate causal inference. We consider continuous and discrete supervised and unsupervised methods for causality inference, and then we introduce a semi-superivsed pairwise probabilistic method. The deep restricted Boltzmann machines and the ways to compute the marginal and conditional probabilities are mentioned before the numerical results. We discuss the results of our experiments on some standard challenges, benchmark networks, and on an original medical problem. Concluding remarks and perspectives close the paper.

2 Related Work

There are two families of causality discovering methods: Additive Noise Models (ANM) and Information Geometric Causal Inference (IGCI) [15].

Additive noise models (ANM) introduced by [7,19] are an attempt to determine causality between two variables. The ANM assume that if there is a function f and some noise E such that $Y = f(X) + E$, where E and X are independent, then the direction is inferred to $X \rightarrow Y$. A generalisation of the ANM, called post-nonlinear models, was introduced by [32]. However, the known drawback of the ANM is that the model is not always suitable for inference on categorical data [3].

Another research avenue exploiting the asymmetry between cause and effect are the linear trace (LTr) method [33] and information-geometric causal inference (IGCI) [8]. They rely on an assumption that if $X \rightarrow Y$, and generating $P(X)$ is independent from $P(Y|X)$, then the trace condition is fulfilled in the causal direction and violated in the opposite one. The IGCI method exploits the fact that the density of the cause and the log slope of the function-transforming cause to effect are uncorrelated. At the same time, the density of the effect and the log slope of the inverse of the function are positively correlated.

Origo [2] is a method to discover causality based on the Kolmogorov complexity. The Minimum Description Length (MDL) principle can be used to approximate the Kolmogorov complexity for real applications. Namely, from algorithmic information viewpoint, if $X \rightarrow Y$, then the shortest program that computes Y from X will be more simple than the shortest program computing X from Y. However, the performance of Origo does not seem to be competitive compared to the ANM.

The bivariate methods are quite different from another state-of-art approach called 3off2 [1] where the algorithm needs three variables to infer a direction, since it considers all possible triplets in data, and looks for colliders in a graph. Therefore, the 3off2 is not suitable for bivariate cases.

A number of recent, reported to be efficient causality discovering methods (see, e.g., [8,9,26]) are based on a postulate of independence of input and output, telling that a causality direction can be inferred from estimated marginal and conditional probabilities of random variables from a data set. In the following, we investigate this research direction.

3 Pairwise Semi-supervised Causal Inference

In this section, we consider methods which rely on the following postulate [8,9,26] and assumptions which are widely used in the domain of bivariate causality inference.

Postulate 1. *If $X \to Y$, then the marginal distribution of the cause $P(X)$ and the conditional distribution of the effect given the cause $P(Y|X)$ are "independent" in the sense that $P(Y|X)$ contains no information about $P(X)$ and vice versa.*

Assumption 1. *We assume that the training procedure has access to N pairs $\{X_i, Y_i\}_{i=1}^{N}$ of observations, and N' points of unlabeled data $\{X_i\}_{i=1}^{N'}$. Let us denote $X = (X_1, \ldots, X_N)$ as a one-dimensional vector, and $Y = (Y_1, \ldots, Y_N)$ is also a vector of length N.*

Assumption 2. *Only X and Y are observed. We assume that no confounders are present, no selection bias, and no feedback.*

Assumption 3. *We formulate the task as a problem of causality inference between two discrete variables, denoted $Y \in \mathcal{Y}$ and $X \in \mathcal{X}$. Without loss of generality, we assume that the causality between them exists, and the main task remains to define what is the cause and what is the effect, i.e. to make a choice between $X \to Y$ and $Y \to X$.*

3.1 Supervised Causal Inference with Inverse Regression

A supervised method of causal inference for two continuous univariate random variables which involves estimation of the conditional probability was proposed by [26]. The theoretical foundation of the CURE (Causal inference with Unsupervised inverse REgression) method relies on the Postulate 1. The asymmetry allows to reduce the problem of the causality inference to the estimation of the conditional probability. More precisely, the CURE method returns $X \to Y$ when the estimation of the conditional probability of cause given effect $P(X|Y)$ based on samples from the marginal probabilities $P(Y)$ is more accurate than the estimation of the conditional probability $P(Y|X)$ based on the samples from the marginal probability $P(X)$. If that is not the case, $Y \rightarrow X$ is inferred.

A way to quantify the accuracy of estimation of $P(X|Y)$ and of $P(Y|X)$, is to analyse the difference between the negative unsupervised log-likelihood and the supervised log-likelihood:

$$D_{X|Y} = \mathcal{L}^{\text{unsup}}_{X|Y} - \mathcal{L}^{\text{sup}}_{X|Y} = \tag{1}$$

$$-\frac{1}{N}\sum_{i=1}^{N} \log p(X_i|Y_i, \mathbf{y}) + \frac{1}{N}\sum_{i=1}^{N} \log p(X_i|Y_i, \mathbf{x}, \mathbf{y}), \tag{2}$$

and

$$D_{Y|X} = \mathcal{L}^{\text{unsup}}_{Y|X} - \mathcal{L}^{\text{sup}}_{Y|X} = \tag{3}$$

$$-\frac{1}{N}\sum_{i=1}^{N} \log p(Y_i|X_i, \mathbf{x}) + \frac{1}{N}\sum_{i=1}^{N} \log p(Y_i|X_i, \mathbf{x}, \mathbf{y}). \tag{4}$$

The decision on the edge orientation in the CURE is taken as follows: if $D_{X|Y} < D_{Y|X}$, then the inferred causal direction is $X \rightarrow Y$, otherwise $Y \rightarrow X$. The obvious weakness of the approach is the high computational complexity, since it relies on the Markov chain Monte Carlo (MCMC) method for the approximation of the posterior distribution, what is computationally consuming in case where the number of samples is large.

3.2 Supervised Causality Discovery with Distance Correlation

Recently, [11] proposed a causality inference method for discrete data. The method is also based on the Postulate 1. Let us assume that X and Y are discrete, and the probabilities $P(X)$ and $P(Y|X)$ are realizations of a variable pair. Since both X and Y are categorical, one can present the probability distributions as tables. As stated by [11], a dependence coefficient between $P(X)$ and $P(Y|X)$ can be used to infer causality direction between variables X and Y, and it was proposed to apply the distance correlation [18]. The dependence measures are defined as follows:

$$D_{Y|X} = \mathcal{D}(P(X), P(Y|X)) \tag{5}$$

$$D_{X|Y} = \mathcal{D}(P(Y), P(X|Y)), \tag{6}$$

where $\mathcal{D}(a, b)$ is the distance correlation.

Given a data set, the distance measures can be computed directly. However, it is not so straightforward to infer causal directions. In was shown by experiments [11] that the correlation distance indeed can be used to characterize the dependence between $P(X)$ and $P(Y|X)$. However, in case where $D_{Y|X}$ is close to $D_{X|Y}$, the causal direction can not be decided.

3.3 Semi-supervised Causal Direction Discovering

In this section, we introduce our method of discovering causal directions. Let $Q(X)$ and $Q(Y)$ be marginal distributions of observations computed from alternative unlabeled, potentially infinite data sets. Here we consider two semi-supervised settings:

1. The distance correlation (Eqs. (5) and (6)) can incorporate unlabeled data in form of $Q(X)$ and $Q(Y)$, and, therefore, hopefully, take into account marginal probabilities which are much more accurate;
2. The difference between a supervised and a semi-supervised log-likelihoods can be a measure that helps to infer causality. In particular, we guess that the parametric functions allow to integrate knowledge about data structure into the criterion, what can be of a big interest in a number of applications.

In this section, we detail the second setting only, since the first scenario based on the distance correlation is straightforward to implement. In our experiments, we consider both settings.

Assumption 4. *We assume that data are discrete or discretized, and the probability distributions can be stocked as two-dimensional (for conditional probability) and one-dimensional (for marginal probability) tables.*

Note that this assumption was also used by [11]. Without loss of generality, the matrices containing the distributions can be computed as follows:

$$p(y|x) = \frac{\sum_{i=1}^{N} \mathbb{1}_{\{X_i=x, Y_i=y\}}}{\sum_{i=1}^{N} \mathbb{1}_{\{X_i=x\}}}, \text{ and } q(x) = \sum_{i=1}^{N'} \mathbb{1}_{\{X'_i=x'\}}. \tag{7}$$

The marginal probability $q(x)$ can be computed from unlabeled data, whose size can potentially be very big.

The semi-supervised criterion where the conditional probability is estimated from labeled data, and the marginal probability can be estimated from numerous unlabeled data takes the following form:

$$p(y|x)q(x) = \frac{\sum_{i=1}^{N} \mathbb{1}_{\{X_i=x, Y_i=y\}}}{\sum_{i=1}^{N} \mathbb{1}_{\{X_i=x\}}} \sum_{i=1}^{N'} \mathbb{1}_{\{X'_i=x'\}}. \tag{8}$$

Note that it was shown that the weighted semi-supervised criterion is asymptotically optimal [29], and it reaches the minimal asymptotic variance.

A way to quantify the accuracy of estimation of $P(X|Y)$ and of $P(Y|X)$, is to compute the difference between the negative *semi-supervised* log-likelihood and the supervised functions:

$$D_{X|Y} = \mathcal{L}_{X|Y}^{\text{semi-sup}} - \mathcal{L}_{X|Y}^{\text{sup}} = \tag{9}$$

$$-\frac{1}{N}\sum_{i=1}^{N} \log p(X_i|Y_i)q(Y_i) + \frac{1}{N}\sum_{i=1}^{N} \log p(X_i|Y_i), \tag{10}$$

and

$$D_{Y|X} = \mathcal{L}^{\text{semi-sup}}_{Y|X} - \mathcal{L}^{\text{sup}}_{Y|X} = \tag{11}$$

$$-\frac{1}{N}\sum_{i=1}^{N} \log p(Y_i|X_i)q(X_i) + \frac{1}{N}\sum_{i=1}^{N} \log p(Y_i|X_i). \tag{12}$$

The decision on the edge directions is similar to the CURE method: if $D_{X|Y} < D_{Y|X}$, then the direction is fixed to $X \rightarrow Y$, otherwise $Y \rightarrow X$. Here, we do not introduce any threshold to be fixed. Indeed, in some cases, where we would like to control the confidence of our decisions, we could introduce a minimal acceptable value which is the difference between $\mathcal{L}^{\text{semi-sup}}$ and $\mathcal{L}^{\text{sup}}$. The pairwise semi-supervised causality inference algorithm is drafted as Algorithm 1.

It is interesting that [25] reported that semi-supervised learning scenario is pointless in general if $P(X)$ contains no information about $P(Y|X)$, i.e. if $X \rightarrow Y$, since a more accurate estimation of $P(X)$ does not influence an estimate of $P(Y|X)$. However, we claim that a more accurate estimation of $P(X)$ would help to infer causality directions more accurately.

Algorithm 1. Semi-Supervised Causal Inference

```
**Input:** Observations $\{X_i, Y_i\}_{i=1}^{N}$, and
unlabeled data $\{X_i\}_{i=1}^{N'}$.

**Output:** Causality directions between $X$ and $Y$

STEP 1: Compute $Q(X)$ and $P(Y|X)$ from data,
Estimate $D_{Y|X} = \mathcal{L}^{\text{semi-sup}}_{Y|X} - \mathcal{L}^{\text{sup}}_{Y|X}$, Eq. 10
(or $D_{Y|X} = \mathcal{D}(P(X), P(Y|X))$, Eq. 5)

STEP 2: Compute $Q(Y)$ and $P(X|Y)$ from data,
Estimate $D_{X|Y} = \mathcal{L}^{\text{semi-sup}}_{X|Y} - \mathcal{L}^{\text{sup}}_{X|Y}$, Eq. 12
(or $D_{X|Y} = \mathcal{D}(P(Y), P(X|Y))$, Eq. 6)

STEP 3: Decide the edge direction:
if $D_{X|Y} < D_{Y|X}$ then
    Infer $X \rightarrow Y$
else
    Infer $Y \rightarrow X$
end if
```

4 Experiments on Benchmark Data Sets

In our experiments, we apply deep restricted Boltzmann machines (DRBM) to estimate the conditional and marginal distributions. A DRBM introduced by [22] contains a set of visible units $\mathbf{v} \in \{0,1\}^D$ and a set of hidden units $\mathbf{h} \in \{0,1\}^P$.

We are in the context of supervised learning, and the DRBM considered in this paper have also output units y; two output units for a binary problem. Energy-based probabilistic models, and the deep RBM, define a probability distribution through an energy function. In the restricted Boltzmann machines, the energy of the state $(\mathbf{v}, \mathbf{h})$ with model parameter w is defined as

$$\mathrm{E}(\mathbf{v}, \mathbf{h}, w) = -\mathbf{v}^T w \mathbf{h}, \tag{13}$$

$$p(\mathbf{v}, w) = \frac{1}{Z(w)} \sum_{\mathbf{h}} \exp(-\mathrm{E}[\mathbf{v}, \mathbf{h}, w]), \tag{14}$$

$$Z(w) = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp(-\mathrm{E}[\mathbf{v}, \mathbf{h}, w]). \tag{15}$$

The conditional distributions over visible and hidden units are given as follows:

$$p(h_j = 1 | \mathbf{v}, \mathbf{h}_{-j}) = \sigma\left(\sum_{i=1}^{D} w_{ij} v_i\right), \tag{16}$$

$$p(v_i = 1 | \mathbf{h}, \mathbf{v}_{-i}) = \sigma\left(\sum_{j=1}^{P} w_{ij} h_j\right), \tag{17}$$

$$\sigma(a) = \frac{1}{1 + \exp(-a)}, \tag{18}$$

and the above defined σ is the logistic function.

All the details for the pre-training and training can be found in [22,23].

Prediction. The prediction is done as follows [16]:

$$p(y|\mathbf{v}) = \sum_{\mathbf{h}} p(y|\mathbf{h}) p(\mathbf{h}|\mathbf{v}) = \mathrm{E}_{p(\mathbf{h}|\mathbf{v})}\, p(y|\mathbf{h}). \tag{19}$$

Computation of the Marginal Probability in the DRBM. Although the conditional distribution of a class given an observation can be directly computed from an estimated deep RBM model, it is not straightforward to compute the marginal probability. To estimate $P(X)$, we use the AIS (Annealed Importance Sampling) algorithm [24] to accurately evaluate the log of the marginal probabilities.

In this section, we illustrate the performance of the proposed approach on standard cause-effect pairs, and on network reconstruction benchmarks. Our implementation is done in Matlab, and it incorporates the publicly available code provided on a web page of Ruslan Salakhutdinov for learning deep Boltzmann machines[1], and for AIS sampling in DRBM[2].

[1] http://www.cs.toronto.edu/~rsalakhu/DBM.html.
[2] https://www.cs.toronto.edu/~rsalakhu/rbm_ais.html.

4.1 Cause-Effect Pairs

We have tested our method on the standard collection of the cause-effect pairs, obtained from http://webdav.tuebingen.mpg.de/cause-effect, version 1.0. The data set contains 100 pairs from different domains, and the ground truth is provided. The goal is to infer which variable is the cause and which is the effect.

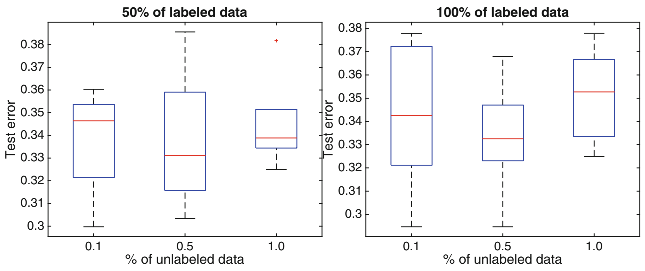


Fig. 1. Experiments on the cause-effect pairs. The accuracy of the semi-supervised criterion based on the log-likelihoods with 50% of training data (on the left), and with 100% of training data (on the right).

The pairs 52–55, 70–71, and 81–83 are excluded from the analysis, since they are multivariate problems. Note that each pair is weighted, and the accuracy is a weighted average.

It was reported that Origo [2] achieves 58% accuracy, and the Additive Noise models (ANM) [19] reach 72 ± 6%. We tested the ANM with the Gaussian Process regression which is the state-of-the-art method. The code is publicly available at http://www.math.ku.dk/~peters/code.html. The goodness of fit is evaluated by the HSIC independence test of the residuals and the input, and the causality inference is based on the obtained p-values for both directions. We observed that the difference between two p-values for almost all cause-effect pairs is close to zero, what provokes the question whether decisions can be reliable.

To compare to the state-of-the-art, we estimate the functional relationships between the cause-effect pairs by the proposed semi-supervised methods. Both settings are developed for discrete data, and we discretize the continuous data using the equal frequency method, the equal width method, and the global equal width method (we use the "infotheo" R package). We also try to find an optimal number of categories for each variable by cross validation, and we test the different number of bins = 3, 5, 7, 10, 15, 20, 25, and 30. We decide to fix the number of bins equal to 5.

Figure 1 illustrates the accuracy of the semi-supervised method based on the log-likelihoods, where we applied Eqs. (10) and (12), as a function of the size of labeled and unlabeled data. We observe that the proposed method achieves the state-of-the-performance. On the left, we see that increasing the size of unlabeled data, we slightly increase the accuracy and also decrease the variance of the error

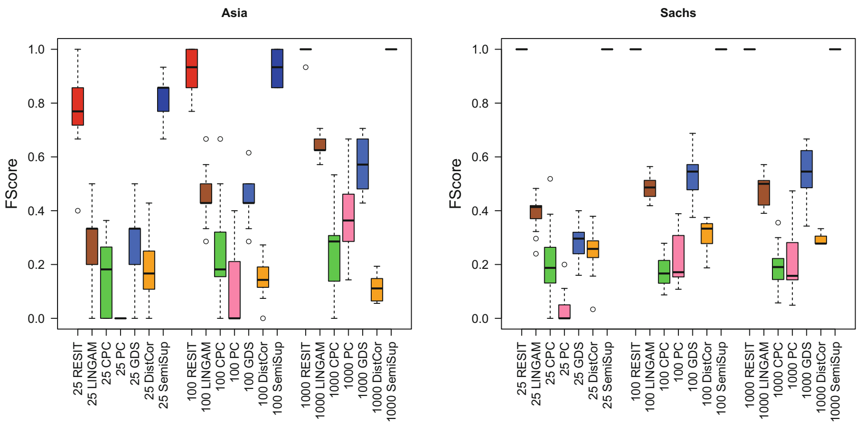


Fig. 2. F-score for the Asia and Sachs data. The number of samples tested is 25, 100, and 1000.

rate. On the right we observe, that some attention is needed while introducing unlabeled data, since in case where we use 100% of labeled and 100% of unlabeled data, it seems that we overfit.

4.2 Network Reconstruction

We run experiments on two benchmark networks, both downloadable from the Bayesian Network Repository[3]:

1. Asia data set [10], also known as the lung cancer benchmark data. The number of nodes is 8, the number of true arcs is 8.
2. Sachs [21] is a causal protein-signaling network with 11 nodes and 17 arcs.

The data sets are network reconstruction challenges with discrete entries. In our experiments, we are interested to discover causality, not the graph structure. We suppose that the skeleton of networks is known, and we compare the causality inference algorithms only.

We test RESIT (regression with subsequent independence test) which is a state-of-the-art ANM method [19]. The RESIT is based on independence tests and simple algorithms that use the independence scores. The algorithm is an iterative procedure where at each iteration, a sink node is identified and disregarded. We also test Linear non-Gaussian Acyclic Model (LiNGAM) approach [28], the PC algorithm named after its inventors Peter Spirtes and Clark Glymour [30], its conservative version CPC [20], and the Greedy DAG search algorithm GDS [6]. The implementation of the state-of-the-art methods mentioned above is publicly available from the web page of Jonas Peters[4].

[3] http://bnlearn.com/bnrepository/.
[4] http://www.math.ku.dk/~peters/code.html.

Table 1. Runtimes for the tested algorithms: RESIT, LiNGAM, CPC, PC, GDS, DC, and the Semi-Supervised approach on Asia and Sachs with 25, 100, and 1000 generated observations.

Method	Asia			Sachs		
	25	100	1000	25	100	1000
Semi-Sup	0.1560	0.1755	0.3005	0.3510	0.7360	1.1350
DC	0.1595	0.1830	0.3160	0.3645	0.7545	1.1480
RESIT	0.1345	0.3345	28.2935	0.5820	57.1550	163.3755
PC	0.0080	0.0100	0.0130	0.0165	0.0530	0.0645
CPC	0.0100	0.0120	0.0215	0.0260	0.1860	0.2455
LiNGAM	1.7600	1.7020	1.7235	0.0875	0.2245	0.3590
GDS	6.0085	590.9255	590.9255	200.7760	8515.4505	8515.4505

The estimated orientations are evaluated for different number of samples. We tested the causality methods with 25, 100, and 1000 observations sampled from the networks. The results are discussed in terms of true positive (TP), false positive (FP) and false negative (FN) edges (i.e. correct, spurious or missing edges respectively). In particular, evaluations are based on Precision = TP/(FP + TP), Recall = TP/(TP + FN), and F-score = 2 * Precision * Recall/(Precision + Recall). We repeat the simulations 10 times, and boxplot the F-scores on Fig. 2. We observe that the RESIT and the proposed semi-supervised method based on the log-likelihoods achieve the best performance. The Distance Correlation (DC) is less efficient.

In real applications, if the network structure is unknown, Aracne (Algorithm for the Reconstruction of Accurate Cellular Networks) can be applied to reconstruct a graph. Aracne introduced by [13] is a state-of-the-art network information-theoretic reconstruction method. The approach defines an edge in a graph as an irreducible statistical dependency. It is reported that the Aracne achieves very low error rates, however, the reconstructed graph is undirected, therefore the Aracne is unable to infer edge directions.

4.3 Runtime Results for Network Reconstruction

We have shown that the proposed algorithm achieves the optimal performance. Another question is its computational efficiency. Table 1 shows the internal time at execution in seconds for different number of tested samples and for different causality methods. The PC and CPC seem to be the fastest to learn but not very accurate. The RESIT has low error rates but its runtimes increase drastically with the number of observations. The semi-supervised method based on the log-likelihoods and the original distance correlation approach need similar time to learn but our method achieves a better F-score. Note that although the proposed algorithm is already quite efficient, the current implementation of our method is not really optimized yet, and it is possible to speed it up.

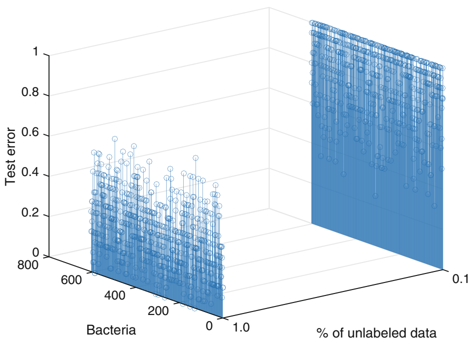


Fig. 3. Metformin and bacteria: test error of the semi-supervised (log-likelihood) causality criterion as a function of the amount of unlabeled data.

5 Effects of Metformin on Human Gut Composition

Recently, a number of associations between chronic human diseases and alterations in gut microbiome composition have been shown [4]. An important question is whether treatment causes changes in human gut flora, and whether it affects metabolism. It was reported [4] that the human gut microbiome of type 2 diabetes is confounded by metformin treatment, and therefore, the drug metformin impacts the composition and richness of the human gut microbiome. Similar results were reported by [31].

The data set of [4] which we explore in our experiments is a multi-country metagenomic dataset, containing information about patients from three countries: Danemark, China, and Sweden. The data contains information of 106 patients with type 2 diabetes who take the metformin, and 93 patients with the diabetes who does not take the drug. The features are 785 gut metagenomes or gut bacteria. The observation matrix contains abundance of bacteria. The abundance matrix is a sparse matrix where 1 means that a metagenome is present in a patient, and 0 means that it is absent.

We run the novel algorithm to test whether it confirms the statements of [4,31] that the metformin alters, in other words, impacts, the gut flora. In the numerical experiments, we assume that the ground truth is that the metformin causes changes in bacteria. If an algorithm predicts the inverse, we consider that it makes an error.

Figure 3 shows that the semi-supervised causality method in generally confirms that the microbiota is affected by the metformin treatment. For the majority of the bacteria considered in the experiments, this relation is obvious with the error rate close or equal to 0. For a few bacteria the accuracy is not so high. However, [4,31] focus on a very limited number of bacteria species, and the statement that the metformin impacts the metagenome is not necessarily true for all bacteria of the human gut flora. Figure 4 illustrates the error rate for one particular bacterium called Akkermansia muciniphila which is associated

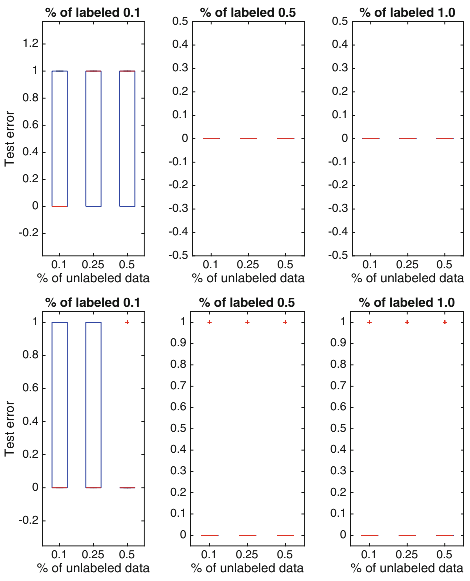


Fig. 4. Metformin and Akkermansia muciniphila: causality prediction error rate as a function of labeled and unlabeled data. Above: the criterion based on the log-likelihoods; below: the setting based on the distance correlation.

with the metabolic health. We clearly see that the hypothesis that the metformin alters the abundance of Akkermansia muciniphila is verified in both proposed semi-supervised settings.

6 Conclusions

We challenged the problem of causal relations discovery from purely observational non-temporal data. In this contribution, we introduced a novel causality inference approach based on a semi-supervised probabilistic framework. The advantage of our approach are its high accuracy, and high computational efficiency. Note that its implementation is simple and straightforward.

We have compared the proposed semi-supervised causality inference algorithm to the state-of-the-art methods, and we illustrate by the experiments on standard data sets and benchmark networks (discrete or discretized data) that the approach achieves the best performance in terms of F-score and accuracy. We have shown that the proposed method is efficient to detect whether a drug causes alterations in the human gut.

From the results of our experiments, we can conclude that measuring distance between a supervised and an unsupervised models can indeed provide information on the edge orientation.

Currently we are investigating another scheme of causality inference which is based on generative hierarchical probabilistic models. Another avenue of research is to extend the proposed method for confounding variables.

Acknowledgments. This work was supported by PEPS (CNRS, France), project MaLeFHYCe.

References

1. Affeldt, S., Verny, L., Isambert, H.: 3off2: a network reconstruction algorithm based on 2-point and 3-point information statistics. BMC Bioinform. **17**(S–2), 12 (2016)
2. Budhathoki, K., Vreeken, J.: Causal inference by compression. In: ICDM (2016)
3. Bühlmann, P., Peters, J., Ernest, J.: CAM: causal additive models, high-dimensional order search and penalized regression. Ann. Stat. **42**, 2526–2556 (2014)
4. Forslund, K., Hildebrand, F., Nielsen, T., Falony, G., Le Chatelier, E., Sunagawa, S., Prifti, E., Viera-Silva, S., Gudmundsdottir, V., Pedersen, H.K., Arumugam, M., Kristiansen, K., Voigt, A.Y., Vestergaard, H., Hercog, R., Costea, P.I., Kultima, J.R., Li, J., Jorgensen, T., Levenez, F., Dore, J., MetaHIT consortium, Nielsen, H.B., Brunak, S., Raes, J., Hansen, T., Wang, J., Ehrlich, S.D., Bork, P., Pedersen, O.: Disentangling the effects of type 2 diabetes and metformin on the human gut microbiota. Nature **528**(7581), 262–266 (2015)
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
6. Hauser, A., Bühlmann, P.: Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. JMLR **13**, 2409–2464 (2012)
7. Hoyer, P., Janzing, D., Mooij, J., Peters, J., Schölkopf, B.: Nonlinear causal discovery with additive noise models. In: NIPS (2009)
8. Janzing, D., Mooij, J., Zhang, K., Lemeire, J., Zscheischler, J., Daniusis, P., Streudel, B., Schölkopf, B.: Information-geometric approach to inferring causal directions. Artif. Intell. **182–183**, 1–31 (2012)
9. Janzing, D., Schölkopf, B.: Causal inference using the algorithmic Markov condition. IEEE Trans. Inf. Theory **56**, 5168–5194 (2010)
10. Lauritzen, S., Spiegelhalter, D.: Local computation with probabilities on graphical structures and their application to expert systems. J. R. Stat. Soc.: Ser. B (Stat. Methodol.) **2**(50), 157–224 (1988)
11. Liu, F., Chan, L.: Causal inference on discrete data via estimating distance correlations. Neural Comput. **28**, 807–814 (2016)
12. Madiraju, A.K., et al.: Metformin suppresses gluconeogenesis by inhibiting mitochondrial glycerophosphate dehydrogenase. Nature **510**, 542–546 (2014)
13. Margolin, A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Favera, R.D., Califano, F.: ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. BMC Bioinform. **7**, S7 (2006)
14. McCreight, L.J., Bailey, C.J., Pearson, E.R.: Metformin and the gastrointestinal tract. Diabetologia **59**, 426–435 (2016)
15. Mooij, J.M., Peters, J., Janzing, D., Zscheischler, J., Schölkopf, B.: Distinguishing cause from effect using observational data: methods and benchmarks. JMLR **17**, 1–102 (2016)
16. Nguyen, T.D., Phung, D., Huynh, V., Lee, T.: Supervised restricted Boltzmann machines. In: UAI (2017)

17. Pearl, J.: Causality: Models, Reasoning and Inference, 2nd edn. Cambridge University Press, Cambridge (2009)
18. Pearson, K.: Notes on the history of correlation. Biometrika **13**, 25–45 (1920)
19. Peters, J., Mooij, J., Janzing, D., Schölkopf, B.: Causal discovery with continuous additive noise models. JMLR **1**(15), 2009–2053 (2014)
20. Ramsey, J., Zhang, J., Spirtes, P.: Adjacency-faithfulness and conservative causal inference. In: UAI (2006)
21. Sachs, K., Perez, O., Pe'er, D., Lauffenburger, D.A., Nolan, G.P.: Causal protein-signaling networks derived from multiparameter single-cell data. Science **308**, 523–529 (2005)
22. Salakhutdinov, R., Hinton, G.: Deep Boltzmann machines. In: AISTATS (2009)
23. Salakhutdinov, R., Larochelle, H.: Efficient learning of deep Boltzmann machines. In: AISTATS (2010)
24. Salakhutdinov, R., Murray, I.: On the qualitative analysis of deep belief networks. In: ICML (2008)
25. Schölkopf, B., Janzing, D., Peters, J., Sgouritsa, E., Zhang, K.: On causal and anticausal learning. In: ICML (2012)
26. Sgouritsa, E., Janzing, D., Hennig, P., Schölkopf, B.: Inference of cause and effect with unsupervised inverse regression. In: AISTATS (2015)
27. Shaw, R.J., et al.: The kinase LKB1 mediates glucose homeostasis in liver and therapeutic effects of metformin. Science **310**, 1642–1646 (2005)
28. Shimizu, S., Hoyer, O., Hyvärinen, A., Kerminen, J.: A linear non-Gaussian acyclic model for causal discovery. JMLR **7**, 2003–2030 (2006)
29. Sokolovska, N., Cappé, O., Yvon, F.: The asymptotics of semi-supervised learning in discriminative probabilistic models. In: ICML (2008)
30. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search. MIT Press, Cambridge (2000)
31. Wu, H., Esteve, E., Tremaroli, V., Khan, M.T., Caesar, R., Manneras-Holm, L., Stahlman, M., Olsson, L.M., Serino, M., Planas-Felix, M., Xifra, G., Mercader, J.M., Torrents, D., Burcelin, R., Ricart, W., Perkins, R., Fernandez-Real, J.M., Backhed, F.: Metformin alters the gut microbiome of individuals with treatment-naive type 2 diabetes, contributing to the therapeutic effects of the drug. Nat. Med. **7**(23), 850–858 (2017)
32. Zhang, K., Hyvärinen, A.: On the identifiability of the post-nonlinear causal models. In: UAI (2009)
33. Zscheischler, J., Janzing, D., Zhang, K.: Testing whether linear equations are causal: a free probability theory approach. In: UAI (2009)

Risk Scores Learned by Deep Restricted Boltzmann Machines with Trained Interval Quantization

Nataliya Sokolovska[1(✉)], Yann Chevaleyre[2], and Jean-Daniel Zucker[3]

[1] Paris Sorbonne University (Paris 6, UPMC), INSERM, Paris, France
nataliya.sokolovska@upmc.fr
[2] University Paris Dauphine, Paris, France
[3] IRD, INSERM, Bondy, Paris, France

Abstract. A compact easily applicable and highly accurate classification model is of a big interest in decision making. A simple scoring system which stratifies patients efficiently can help a clinician in diagnostics or with the choice of treatment. Deep learning methods are becoming the preferred approach for various applications in artificial intelligence and machine learning, since they usually achieve the best accuracy. However, deep learning models are complex systems with non-linear data transformation, what makes it challenging to use them as scoring systems. The state-of-the-art deep models are sparse, in particular, deep models with ternary weights are reported to be efficient in image processing. However, the ternary models seem to be not expressive enough in many tasks. In this contribution, we introduce an interval quantization method which learns both the codebook index and the codebook values, and results in a compact but powerful model.

We show by experiments on several standard benchmarks that the proposed approach achieves the state-of-the-art performance in terms of generalizing accuracy, and outperforms modern approaches in terms of storage and computational efficiency. We also consider a real biomedical problem of a type 2 diabetes remission, and discuss how the trained model can be used as a predictive medical score, and be helpful for physicians.

1 Introduction

Medical doctors and clinicians rely more and more often on artificial intelligence and machine learning tools for diagnostic purposes. The application domains vary from prediction of a risk of a disease to understanding genetic variations.

Deep learning methods are able to learn complex internal representations with the aim to solve complex real problems, and can be applied to any type of medical data, not only images [7]. High-level representations can be learned in an unsupervised or semi-supervised way within a deep learning framework, taking only a small amount of labeled data into account. Top-down feedback in deep networks helps to deal better with uncertainty and to treat noisy data in a more robust way. However, a newly learned high-level data representation is not necessary easily interpretable for clinicians and physicians. Moreover, the networks become deeper and deeper, the number of features increases, and the models are black boxes for human experts.

A *scoring system* stratifies an observation based on an estimated model. Clinical scoring systems are of particular interest since they are expected to predict a state of a patient, and to help physicians to provide accurate diagnostics. Some widely used medical scores are SAPS I, II, and III [5,16] and APACHE I, II, III to assess intensive care units mortality risks [12], CHADS$_2$ to assess the risk of stroke [4]; TIMI to estimate the risk of death of ischemic events [1]. A detailed example of a clinical score, shown in Table 1, is the DiaRem score [25] which is a preoperative method to predict remission of type 2 diabetes after a gastric bypass surgery. The DiaRem is based on four clinical variables and a few thresholds per variable. Only one arithmetic operation is involved into the DiaRem computation: the scores are added, and if the sum is <7, then a patient is likely to benefit from the surgery, and to get the diabetes remission.

Recently, [27] considered a problem of learning *risk scores*, where the proposed models are designed for *risk assessment*. Under the risk assessment it is meant that we model a prediction risk, e.g. using a probabilistic model such as logistic regression. The modelled conditional probability is equal to the risk of having this or that syndrome, or a disease.

We are motivated to apply deep learning to train a scoring system optimized for the risk, since deep learning models achieve high accuracy. On the other hand, only an easily interpretable and easily computable score can be adopted in clinical routines. If a model to provide a prognosis for a new patient is too complex, an application with an intuitive graphical interface is needed for practical use by therapists. In this case, the computations are expected to be fast and efficient, so that such an application can be installed on a small portable gadget.

Several methods have been recently proposed to reduce the size of networks, and, therefore, to increase the computational speed and to reduce the storage needed for deep models. The state-of-the-art approaches rely mostly on reduction of the weights of deep networks, activation units, and event gradients to binary or ternary values [3,13].

Our contribution is multifold:

– A deep model can be huge and the inference can be computationally expensive, and take unacceptably much time, especially if high-dimensional "omics" (transcriptomics, proteomics, etc.) data are integrated into the model. Here, we introduce *deep restricted Boltzmann machines (DRBM) where the weights are quantized by intervals* what drastically reduces the size of the model, since it is sufficient to store the estimated codebook index and the corresponding trained codebook values. The weights quantization also plays a role of a regularizer. Our approach results in a more expressive and more accurate model compared to the state-of-the-art methods with ternary weights.

- The output of the deep restricted Boltzmann machine is probabilistic, what is of a big help to assess the confidence of a decision. We propose to use the DRBM as a probabilistic model to *design the predicted risk* under the form of a conditional probability. The scores distribution can be visualized to simplify the interpretation of prediction. Therefore, our scoring systems are optimized risk scores models which implement the interval quantization for efficient computation and sparse models.
- We test the novel interval quantization DRBM on a number of publicly available data sets, and we show that the reduced compact model achieves an optimal performance in terms of accuracy. We also consider an original problem of type 2 diabetes remission, and discuss how the proposed risk scores can provide additional diagnostics, and help physicians in decision making.

Table 1. The DiaRem score to assess the outcome of the bariatric surgery [25]

Clinical variables	Age				Glycated hemoglobin				Insuline		Other drugs	
Thresholds	<40	40–49	50–59	>60	<6.5	6.5–6.9	7–8.9	>9	Yes	No	Yes	No
Scores	0	1	2	3	0	2	4	6	10	0	3	0

This paper is organised as follows. Related work discusses the state-of-the-art discrete deep learning methods, and the modern approaches to construct scores. Then we consider the deep restricted Boltzmann machines which are the computational framework of our model. We introduce the deep restricted Boltzmann machines where the weights are quantized by intervals. We show the results of our numerical experiments, and the obtained scores. Concluding remarks and perspectives close the paper.

2 Related Work

We are interested in stochastic neural networks whose activation units have a probabilistic element. Such a choice is motivated by the fact that probabilities can also be interpreted, e.g., probability to be ill or to be healthy is an important piece of information. Deep restricted Boltzmann machines (DRBM) originally introduced by [19] is a deep stochastic model with one layer of visible units and several hidden units. The model is restricted in the sense that no visible unit is connected to any other visible unit, and similarly for all hidden layers.

Systems with potentially many layers of non-linear processing such as deep RBM, are difficult to learn, and some approximate inference methods (see e.g., [20–22]) were proposed. The deep RBM were reported to be efficient on a number of various real applications such as natural language processing [24], neuroimaging applications [9], speech processing [28], and also for heterogeneous data integration [23].

Since recently, the artificial intelligence and machine learning communities work actively on the problem to reduce the demand for memory usage by deep architectures that can potentially be huge. Among the state-of-the-art methods to reduce the size of neural networks are approaches which binarize the weights, activation units, and even gradients of the deep networks. Weights quantization appears to have several advantages. During training, most of computational power is spent on multiplications. Quantization of weights removes multiplications in the forward pass, since sampling an integer is faster than multiplication. The sampling process is done once for each mini-batch. Another advantage to have a model with discrete weights, is the regularizing effect preventing from overfitting what is a real problem in deep learning.

A number of quantized deep learning approaches were proposed very recently. So, [3] proposed a neural network with binary weights and binary activations at run-time. During the training procedure, most arithmetic operations are replaced with bit-wise operations, and the approach is, therefore, computationally extremely efficient. Similar idea is implemented in [11] where all the inputs, weights, biases, hidden units, and outputs can be represented with single bits. [15] introduced quantized back propagation where a considerable number of multiplications is eliminated from the backward pass. A quite efficient scheme to binarize weights, called BinaryConnect, was introduced by [2], and [13] reported that expressive ability of ternary networks, i.e., of models where weights take their values in $\{-1, 0, 1\}$, is better than of binary models. Deep neural networks with the ternary weights were also designed by [10]. [29] considered training in the ternary networks, and proposed to use scaling coefficients for each layer to increase the expressive power of the ternary models.

It is important to note that the introduced quantized deep learning methods leading to much more compact models, do not degrade performance compared to real-valued models.

Learning *medical scores* is a rather new domain of research, and the extensive literature does not really exist. Among the state-of-the-art methods are Supersparse Linear Integer Models (SLIM) developed by [26] for automated medical score learning, which are reported to perform efficiently on various data sets. The model is formulated as an integer programming task and optimizes directly the accuracy, the 0–1 loss, and the degree of sparsity. Similarly to [27], our model is designed for risk assessment, however, in contrast to it, the optimization procedure is quite different from the riskSLIM which is based on a cutting plane algorithm and MIP solvers.

3 Problem Statement

We define the problem of scoring systems learning as follows. We have a set of training examples $\{X_i, Y_i\}_{i=1}^N$, where X is a matrix of observations, and Y is a class label. A score function is defined as $\langle \theta, X \rangle$, where θ is a coefficient vector, and $\langle \cdot, \cdot \rangle$ is the scalar product. Given observations X, and estimated weights θ, a score s_i for an observation X_i is equal to $\langle \theta, X_i \rangle$. A class can be predicted according to the conditional probability

$$p(y=1|X) = \frac{1}{1+\exp(-\langle\theta, X\rangle)}. \tag{1}$$

The conditional probability designs also the *predicted risk*, and we hope that it can be estimated with a deep learning approach more accurately than with the standard logistic regression.

4 Deep Restricted Boltzmann Machines

A deep restricted Boltzmann machine (DRBM) introduced by [19] contains a set of visible units $\mathbf{v} \in \{0,1\}^D$, and a set of hidden units $\mathbf{h} \in \{0,1\}^P$. We are in the context of supervised learning, and the DRBM considered in this paper have also output units y; two output units for a binary problem. Energy-based probabilistic models, and the deep RBM, define a probability distribution through an energy function. In the restricted Boltzmann machines, the energy of the state $(\mathbf{v}, \mathbf{h})$ with model parameter w is defined as

$$\mathrm{E}(\mathbf{v}, \mathbf{h}, w) = -\mathbf{v}^T w \mathbf{h}, \quad p(\mathbf{v}, w) = \frac{1}{Z(w)} \sum_{\mathbf{h}} \exp(-\mathrm{E}[\mathbf{v}, \mathbf{h}, w]), \tag{2}$$

$$Z(w) = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp(-\mathrm{E}[\mathbf{v}, \mathbf{h}, w]). \tag{3}$$

The conditional distributions over visible and hidden units are given as follows:

$$p(h_j = 1|\mathbf{v}, \mathbf{h}_{-j}) = \sigma(\sum_{i=1}^{D} w_{ij} v_i), \quad p(v_i = 1|\mathbf{h}, \mathbf{v}_{-i}) = \sigma(\sum_{j=1}^{P} w_{ij} h_j), \tag{4}$$

where $\sigma(a) = \frac{1}{1+\exp(-a)}$, and the above defined σ is the logistic function. The gradient to run an optimization procedure can be written as

$$\Delta w = \alpha(\mathrm{E}_{Pdata}[\mathbf{v}\mathbf{h}^T] - \mathrm{E}_{Pmodel}[\mathbf{v}\mathbf{h}^T]), \tag{5}$$

where α is the learning rate, the first term is the expectation with respect to the completed data distribution, and the second term is the expectation with respect to the distribution defined by the model.

If we consider a two-layer deep restricted Boltzmann machine, the energy of state is given by

$$\mathrm{E}[\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2, w] = -\mathbf{v}^T w^1 \mathbf{h}^1 - \mathbf{h}^1 w^2 \mathbf{h}^2, \tag{6}$$

where $w = \{w^1, w^2\}$ are the parameters of the model, and

$$p(\mathbf{v}, w) = \frac{1}{Z(w)} \sum_{\mathbf{h}^1, \mathbf{h}^2} \exp(-\mathrm{E}[\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2, w]). \tag{7}$$

The conditional distributions over the hidden and visible layers are given as follows:

$$p(h_j^1 = 1|\mathbf{v}, \mathbf{h}^2) = \sigma(\sum_i w_{ij}^1 v_i + \sum_m w_{jm}^2 h_j^2), \tag{8}$$

$$p(h_m^2 = 1|\mathbf{h}^1) = \sigma(\sum_i w_{im}^2 h_i^1), \quad p(v_i = 1|\mathbf{h}^1) = \sigma(\sum_j w_{ij}^1 h_j^2). \tag{9}$$

Pretraining. To initialize the weights w of the model, we perform the greedy layerwise pretraining [8]. The greedy layerwise pretraining learns a stack of restricted Boltzmann machines in an unsupervised layer-by-layer greedy procedure. It was shown by [8,19] that such a pretraining initializes the weights to reasonable values and therefore fastens the approximate inference to estimate the model. To perform the initialization, we compute:

$$p(h_j^1 = 1|\mathbf{v}) = \sigma(\sum_i w_{ij}^1 v_i + \sum_i w_{ij}^1 v_i), \quad p(v_i = 1|\mathbf{h}^1) = \sigma(\sum_j w_{ij}^1 h_j), \tag{10}$$

$$p(h_j^1 = 1|\mathbf{h}^2) = \sigma(\sum_m w_{jm}^2 h_m^2 + \sum_m w_{jm}^2 h_m^2), \quad p(h_m^2 = 1|\mathbf{h}^1) = \sigma(\sum_j w_{jm}^2 h_j^1), \tag{11}$$

where the input is doubled to eliminate the double-counting problem while top-down and bottom-up inferences are combined. When the Eqs. (10)–(11) are combined, we get

$$p(h_j^1 = 1|\mathbf{v}, \mathbf{h}^2) = \sigma(\sum_i w_{ij}^1 v_i + \sum_m w_{jm}^2 h_m^2) \tag{12}$$

Training. Let

$$F(\mathbf{v}) = -\log \sum_{\mathbf{h}^1, \mathbf{h}^2} \exp(-E[\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2]), \tag{13}$$

and

$$\frac{-\partial \log p(\mathbf{v}, w)}{\partial w} = \frac{\partial F(\mathbf{v})}{\partial w} - \sum_{\tilde{v}} p(\tilde{v}) \frac{\partial F(\tilde{v})}{\partial w}, \tag{14}$$

where the first term increases the probability of training data and is often referred to as the positive phase, and the second term decreases the probability of samples generated by the model and is associated with the negative phase. As we have already mentioned earlier, the second term of the derivative is an expectation over all possible configurations of input, and its computation is usually intractable. However, it can be computed using sampling

$$\mathrm{E}_p \left[\frac{\partial F(\mathbf{v})}{\partial w} \right] = \frac{1}{|\mathcal{V}|} \sum_{\bar{v} \in \mathcal{V}} \frac{\partial F(\bar{v})}{\partial w}, \tag{15}$$

where $\bar{v} \in \mathcal{V}$ are samples produced, e.g., by a MCMC method. Annealed Importance Sampling (AIS) with variational inference can be used to make the computations tractable (see [19,22] for details).

Prediction. The prediction is done as follows [17]:

$$p(y|\mathbf{v}) = \sum_{\mathbf{h}} p(y|\mathbf{h})p(\mathbf{h}|\mathbf{v}) = \mathrm{E}_{p(\mathbf{h}|\mathbf{v})}\, p(y|\mathbf{h}). \tag{16}$$

5 Quantized Deep Restricted Boltzmann Machines

Binary and ternary connects are nowadays the state-of-the-art utilities to discretize deep networks. In the following, w^b are weights of a deep model obtained with the binary connect, and w^t is the result of the ternary connect transformation.

Binary connect [3] transforms each full precision element of w into an integer in $\{-1, 1\}$ as follows:

$$w^b \sim \mathrm{Bernoulli}(\frac{w+1}{2}) * 2 - 1. \tag{17}$$

In the DoReFa approach, it is proposed to quantize weights, activations, and gradients using various widths of bits. To quantize the weights, the average of absolute values of full precision weights are taken layerwise:

$$w^b = \mathrm{E}(|w|) * \mathrm{sign}(w). \tag{18}$$

In practice, a lot of w are close to zero, and it is usually beneficial to allow weights to be zero. Ternary connect [13] does another simple transformation of w to map them to values in $\{-1, 0, 1\}$:

$$w^t \sim \mathrm{Bernoulli}(|w|) * \mathrm{sign}(w). \tag{19}$$

The back propagation pass is also quantized [15], and during the back propagation most of the floating point multiplications are also eliminated.

During the back propagation

$$\frac{\partial F}{\partial w} = \frac{\partial F}{\partial w^b} = \frac{\partial F}{\partial w^t}. \tag{20}$$

Trained ternary quantization [29] relies on precision coefficients W_h^p and W_h^n for each layer h, and the weights take one of three possible values $\{W_h^p, 0, W_h^n\}$ instead of $\{-1, 0, 1\}$. The gradient is back propagated for both W_h^p and W_h^n, and also for the full precision weights.

Potentially, a model where $W_h^p \neq -W_h^n$ is more expressive, and the learned coefficients play the role of "learning rate multipliers". To learn the ternary weights, where the coefficients W_h^p and W_h^n are trained using back propagation, [29] proposed the following procedure. Quantization values W_h^p and W_h^n are

Algorithm 1. Trained Interval Quantization in DRBM

```
Input: $X$, $Y$, number of layers and units, learning rate $\alpha$
Output: Quantized $\bar{w}$ and predictive scores

// for some number of epochs T
for for t = 1 : T do
    // Forward pass
    for each layer $h$ do
        Compute activations from $w$, $h^{-1}$, and $b$
        Quantize $w$ and get $\bar{w}^{+}_{h,k^+}$ and $\bar{w}^{-}_{h,k^-}$ with eq. (25) and eq. (26)
    end for
    // Backward pass
    for each layer $h$ do
        Compute $\frac{\partial F}{\partial W^{+}_{h,k^+}}$ with eq. (27) and compute $\frac{\partial F}{\partial W^{-}_{h,k^-}}$ with eq. (28)
        Compute $\frac{\partial F}{\partial w^{+}_{h}}$ with eq. (29) and compute $\frac{\partial F}{\partial w^{-}_{h}}$ with eq. (30)
        Update weights, both continuous $w$ and the coefficients $W^{+}_{h,k^+}$ and $W^{-}_{h,k^-}$
    end for
end for
```

introduced for each layer h; Δ_h is a thresholds for layer h. During the forward pass, the ternary weights are computed as follows:

$$w_h^t = \begin{cases} W_h^p, \text{ if } w_h > \Delta_h, \\ 0, \text{ if } |w_h| \leq \Delta_h, \\ -W_h^n, \text{ if } w_h < -\Delta_h. \end{cases} \tag{21}$$

The coefficients W_h^p and W_h^n are independent and trained during the training procedure. The gradients can be directly computed as follows:

$$\frac{\partial F}{\partial W_h^p} = \sum_{i \in I_h^p} \frac{\partial F}{\partial w_h^t(i)}, \tag{22}$$

$$\frac{\partial F}{\partial W_h^n} = \sum_{i \in I_h^n} \frac{\partial F}{\partial w_h^t(i)}, \tag{23}$$

where $I_h^p = \{i | w_h(i) > \Delta_h\}$ and $I_h^n = \{i | w_h(i) < -\Delta_h\}$. In the presence of these two scaling factors, the gradient of the full precision weights w takes the following form:

$$\frac{\partial F}{\partial w_h} = \begin{cases} W_h^p \times \frac{\partial F}{\partial w_h^t} \text{ if } w_h > \Delta_h, \\ 1 \times \frac{\partial F}{\partial w_h^t} \text{ if } |w_h| \leq \Delta_h, \\ W_h^n \times \frac{\partial F}{\partial w_h^t} \text{ if } w_h < -\Delta_h. \end{cases} \tag{24}$$

6 Trained Interval Quantization

The main motivation to quantize weights by intervals in more than three bins, i.e. to have a model with more than $\{W_h^n, 0, W_h^p\}$ coefficients, is to increase the expressiveness of the model. Ternary quantization, even with W_h^p and W_h^n learned and different for each layer, is not powerful enough for a number of applications. In this section, we describe the interval quantization where the number of values, which can be taken by the parameters, is bigger than three. We introduce coefficients W_{h,k^+}^+ and W_{h,k^-}^- associated with positive and negative weight intervals $k^+ \in K^+$ and $k^- \in K^-$ of layer h. The coefficients are learned during the training procedure, similarly to the method of [29].

For each layer h of the network, we discretize the positive part w_h^+ of the parameter vector w_h into K_h^+ bins. The negative part w_h^- of the parameter vector w_h of layer h is split into K_h^- bins. Although the choice of a binning algorithm can be important, in this paper we consider two standard binning approaches, namely, the equal width and the equal frequency binning. From a number of numerical results, we decided to focus on the equal width algorithm only. So, first, the equal width binning approach cuts w_h into $\bar{w}_{h,k^+}^+$ and $\bar{w}_{h,k^-}^-$, and during the forward pass:

$$\bar{w}_{h,k^+}^+ = W_{h,k^+}^+, \text{ for each bin } k^+ \text{ from } K_h^+ \text{ positive bins,} \tag{25}$$

$$\bar{w}_{h,k^-}^- = -W_{h,k^-}^-, \text{ for each bin } k^- \text{ from } K_h^- \text{ negative bins.} \tag{26}$$

The gradients with respect to the coefficients W_{h,k^+}^+ and W_{h,k^-}^- which are associated with the positive and negative bins and which are needed during the backward procedure:

$$\frac{\partial F}{\partial W_{h,k^+}^+} = \sum_{i \in I_{h,k^+}} \frac{\partial F}{\partial \bar{w}_{h,k^+}^+(i)} \text{ for all } k^+, \tag{27}$$

$$\frac{\partial F}{\partial W_{h,k^-}^-} = \sum_{i \in I_{h,k^-}} \frac{\partial F}{\partial \bar{w}_{h,k^-}^-(i)} \text{ for all } k^-, \tag{28}$$

where $I_{h,k^+} = \{i | w_{h,k^+}^+ \text{ for all } k^+ \in K^+\}$, and $I_{h,k^-} = \{i | w_{h,k^-}^- \text{ for all } k^- \in K^-\}$. In the presence of these multiple scaling factors, the gradient of the full precision weights w_h, for the positive w_h^+ and negative w_h^- parts, can be written as follows:

$$\frac{\partial F}{\partial w_h^+} = W_{h,k^+}^+ \times \bar{w}_{h,k^+}^+ \text{ for all } k^+ \in K^+, \tag{29}$$

$$\frac{\partial F}{\partial w_h^-} = W_{h,k^-}^- \times \bar{w}_{h,k^-}^- \text{ for all } k^- \in K^-. \tag{30}$$

The complete procedure to learn the deep Boltzmann machines with the interval quantization is drafted as Algorithm 1.

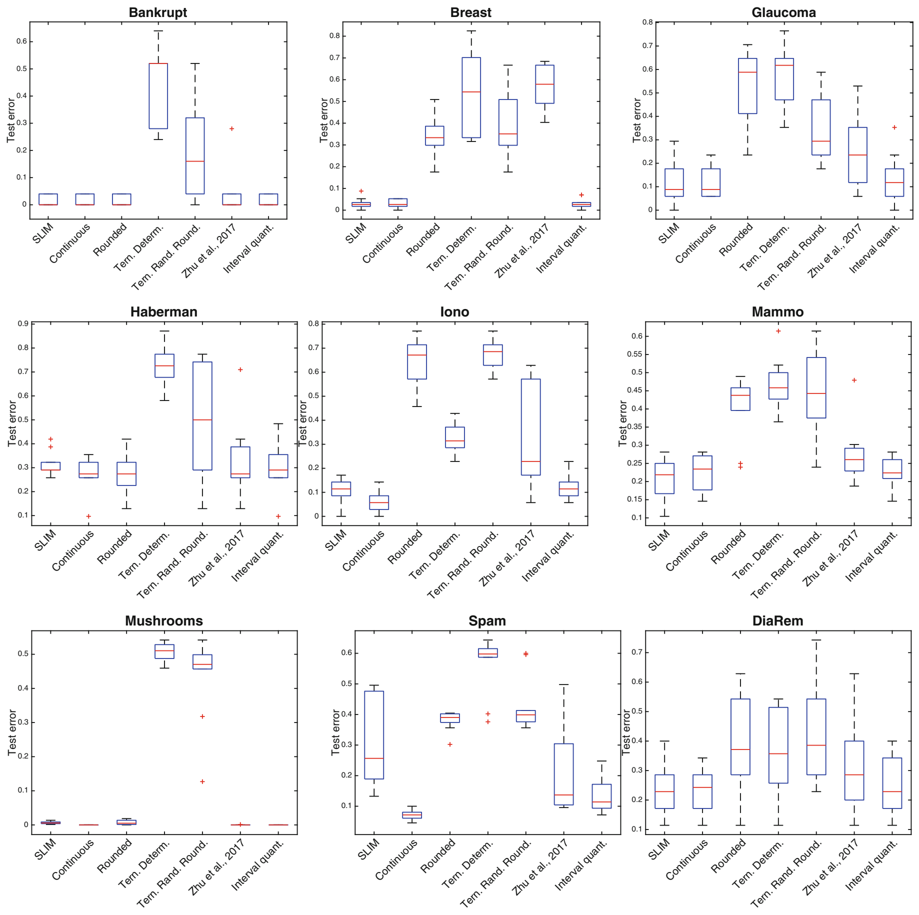


Fig. 1. Performance of the SLIM, Continuous DRBM, DRBM with rounded weights, DRBM with ternary deterministic weights, DRBM with ternary randomized rounding weights, DRBM with trained ternary weights, and DRBM with the interval quantization.

In the numerical experiments, to project the full precision values of w into a given interval $[-R; R]$ we use a method described in [6]. A randomized rounding is used in our experiments to produce the ternary weights; this is one of baseline methods we run to compare our results with. Our implementation is done in Matlab, and it is based on the publicly available code provided on a web page of Ruslan Salakhutdinov for learning deep Boltzmann machines[1].

[1] http://www.cs.toronto.edu/~rsalakhu/DBM.html.

7 Experiments

In this section, we share our results on several standard benchmarks, and on an original problem of type 2 diabetes remission prediction after a gastric bypass surgery. The choice of the benchmark data sets is motivated by the results of the scoring systems learned by the SLIM and reported in [26]. Most of the sets are downloadable from the UCI Machine Learning repository[2] [14]:

- Qualitative Bankruptcy. The data are categorical, with 250 observations and 7 attributes such as industrial risk, management risk, competitiveness, etc.
- Breast Cancer Wisconsin (Prognostic). We dispose of about 30 parameters describing characteristics of the cell nuclei present in the medical images for 198 patients. All parameters are continuous.
- Ionosphere. Ionosphere data set contains 351 observations and 34 attributes to discriminate good and bad ionosphere conditions. The observations are radar data, and all of them are continuous.
- Mammography. The Mammography set contains 961 observations and 6 variables, and the goal is to predict the outcome of the breast cancer screening, and to avoid unnecessary invasive procedures such as biopsy.
- Haberman's Survival data. The aim is to classify patients according to the survival outcome after surgery for breast cancer. There are only three attributes (age of patient, year of operation, and the number of positive auxiliary nodes detected) for 306 patients.
- Mushrooms. The data set is discrete, the number of instances is 8124, and the number of attributes is 22. The goal is to predict whether mushrooms are poisonous or edible.
- Spambase. The set contains 4601 emails which can be divided into two categories (spam or not spam). All 57 attributes are either continuous (normalized word frequencies) or integers (number of capital letters, longest sequence of capital letters, etc.)
- Glaucoma. The Glaucoma diagnosis set includes data from laser scanning images taken from the eye background for 170 patients and 66 attributes, providing information on the morphology of the optic nerve head, the visual field, the intra ocular pressure and a membership variable. The data is part of the "ipred" R package [18].
- Prediction of Diabetes Remission. The data set of type 2 diabetic subjects is produced and managed by the Department of Nutrition, Center of Reference for Medical and Surgical Care of Obesity, at the Institute of Cardiometabolism and Nutrition (ICAN), Pitié-Salpêtrière Hospital (Paris, France). About 130 type 2 diabetes patients were recruited, and they are characterized by 6 parameters. The challenge is to predict whether a patient will have a remission after a gastric bypass surgery or not.

Figure 1 demonstrates the test error rates for all considered data sets. We compare the performance of our approach ("Interval quant." on Fig. 1) with the

[2] http://archive.ics.uci.edu/ml/.

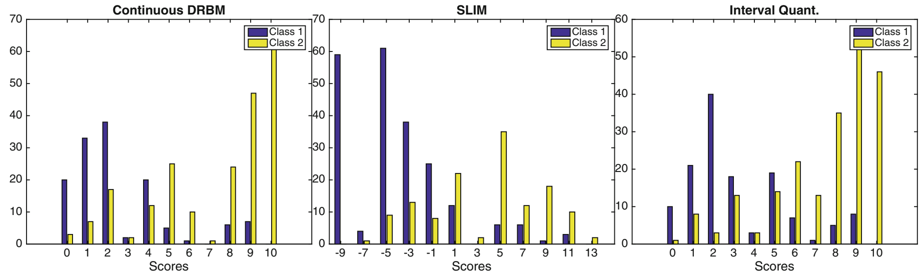


Fig. 2. The distributions of the obtained scores on the DiaRem Data the number of observations for each possible score value. On the left: the continuous DRBM; in the center: the SLIM, and on the right: the DRBM quantized by interval.

following methods: (1) SLIM of [26], since this approach is the state-of-the-art method to learn scoring systems; (2) trained ternarization of [29]; (3) standard continuous DRBM to control that the discretized models do not perform worse than the state-of-the-art ("Continuous" on Fig. 1); (4) DRBM with rounded weights; (5) DRBM with ternary weights, i.e. the weights are projected into the interval $[-1; 1]$, and rounded (deterministically); (6) DRBM with ternary weights obtained by the randomized rounding.

We perform 10-fold cross validation and boxplot the testing error. The number of epochs for training is fixed to 75. We verified that the optimization procedure converged. We run a number of preliminary tests to set the following DRBM configuration: 4 hidden layers with 100 units in each. To bin the weights, we apply the equal width binning with the histcounts() Matlab function which partitions a vector into bins. We do not fix the number of bins a priori in the function.

The standard deep RBM achieves an excellent error rate but the continuous model is complex. The SLIM scoring system also reaches a very competitive performance. Using the SLIM approach, we got the same results as reported by [26]. The only discrepancy with [26] concerns the Spam data: the authors of SLIM report the error rate $6.3\% \pm 1.2$, and we using SLIM get a result which is significantly worse. It can be related to the choice of the hyperparameters, although we fixed the hyperparameters using 10-fold cross validation. The model with a posteriori rounded weights, and the models with ternary weights both randomly and deterministically do not perform well. The ternary model of [29] where the coefficients are trained and are different from -1 and $+1$, achieves the state-of-the-art accuracy on some data sets. In general, the models where the weights can take three values only seem to be not expressive enough. The proposed interval quantization reaches the optimal performance.

Table 2 provides the number of unique values (the codebook values) per layer of the DRBM trained with the interval quantization. Note that the method of [29] keeps 3 values only, and although the method was reported to be very promising on images, it is not so stable on other data. The standard continuous DRBM keeps in our case thousands (20, 000–30, 000 depending on a data set) of parameters, and the novel interval quantization has a flexible number of parameters per layer, the model is quite compact, and it reaches the same performance as the SLIM and the continuous DRBM. The models discretized by the interval quantization are about 100 times more compressible than the continuous DRBM.

Table 2. The number of unique values in each layer of the deep network trained with the interval quantization. L stands for "layer".

	L 1	L 2	L 3	L 4		L 1	L 2	L 3	L 4
Bankrupt	38	114	122	19	Mammo	35	106	99	16
Breast	57	117	135	14	Mushrooms	124	136	109	10
Glaucoma	93	82	84	19	Spam	85	128	93	10
Haberman	19	101	79	16	DiaRem	29	127	147	19
Iono	56	147	115	12					

Another important aspect of scoring systems are the obtained scores, and their simplicity. A risk value which is a conditional probability, is an indicator of a state of a patient. The risk value can also be taken into consideration by physicians, it can be rounded or scaled for simplicity. The risk scores issued from the DRBM are scaled conditional distributions of classes given observations. In the SLIM, a score is a linear combination of model parameters and an observation.

Figure 2 shows the scores learned from the Diabetes Remission data. We show the scores learned with the SLIM, with the continuous DRBM, and with the interval quantization. The scores obtained with the method of [29] are not informative, and the corresponding accuracy is low. We see that the error is different from zero for all methods, and we always observe some overlap between the classes, what is conform with the state-of-the-art performance which is about 82% accuracy. The scores produced by the SLIM model (e.g., Fig. 2 in the center) can lie on an arbitrary interval $(-\infty; +\infty)$, and are less interpretable than probability values for clinicians and therapists. However, it is also possible to compute the conditional probabilities from an estimated SLIM model.

The risk scores of the interval quantization method are the conditional probabilities multiplied by 10 what is an arbitrary constant. If we look at the scores distribution produced by the interval quantization method, we will notice that patients with the scores equal to 0, 1, and 2 will benefit from the gastric bypass surgery with a high probability. At the same time, the patients with the scores 7–10 will probably not have the remission. We are not confident in our decision for the patients who are in the interval 3–6. Such kind of information is very helpful for clinicians in decision making, in choice of treatment, and in their discussions with patients.

8 Conclusion

In this contribution we have tried to address the problem of accurate risk score learning from rich continuous data in a computationally efficient way. We proposed a novel quantized deep restricted Boltzmann machines, and our main result is presented as Algorithm 1, where we learn simultaneously the codebook index (the quantization intervals), and the corresponding codebook values (the coefficients associated with the bins). We claim that the conditional probabilities of a class given an observation can provide an important information on the confidence of the prediction, and, therefore, can be used as a risk score.

We have shown by experiments on several benchmarks that the proposed method of the interval quantization is promising, competitive, and outperforms the state-of-the-art scoring systems and classifiers in terms of computational efficiency, and is highly accurate. Another important result on real medical data is described in the experimental section. We illustrated by the diabetes remission problem the potential of the proposed approach to efficiently learn risk scores, what traditionally costs many hours of work of human experts.

Currently we are investigating how to perform adaptive interval quantization, and how to adopt other deep architectures to learn scoring systems from heterogeneous data.

Acknowledgments. This work was supported by PEPS (CNRS, France), project MaLeFHYCe, and by the French National Research Agency (ANR JCJC DiagnoLearn).

References

1. Antman, E., Cohen, M., Bernink, P., McCabe, C., Horacek, T., Papuchis, G., Mautner, B., Corbalan, R., Radley, D., Braunwald, E.: The TIMI risk score for unstable angina/non-ST elevation MI. J. Am. Med. Assoc. **284**, 835–842 (2000)
2. Courbariaux, M., Bengio, Y., David, J.-P.: Binaryconnect: training deep neural networks with binary weights during propagations. In: NIPS (2015)
3. Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized neural networks. In: NIPS (2016)
4. Gage, B.F., Waterman, A.D., Shannon, W., Boechler, M., Rich, M.W., Radford, M.J.: Validation of clinical classification schemes for predicting stroke. J. Am. Med. Assoc. **285**, 2864–2870 (2001)
5. Le Gall, J.-R., Lemeshow, S., Saulnier, F.: A new simplified acute physiology score (SAPS II) based on a European/North American multicenter study. J. Am. Med. Assoc. **270**, 2957–2963 (1993)
6. Golovin, D., Sculley, D., McMahan, H.B., Young, M.: Large-scale learning with less RAM via randomization. In: ICML (2013)
7. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
8. Hinton, G., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)

9. Hjelma, D., Calhouna, V., Salakhutdinov, R., Allena, E., Adali, T., Plisa, S.: Restricted Boltzmann machines for neuroimaging: an application in identifying intrinsic networks. NeuroImage **96**, 245–260 (2014)
10. Hwang, K., Sung, W.: Fixed-point feedforward deep neural network design using weights +1, 0, and −1. In: SiPS (2014)
11. Kim, M., Smaragdis, P.: Bitwise neural networks. In: ICML Workshop on Resource-Efficient Machine Learning (2015)
12. Knaus, W.A., Zimmerman, J.E., Wagner, D.P., Draper, E.A., Lawrence, D.E.: APACHE-acute physiology and chronic health evaluation: a physiologically based classification system. Crit. Care Med. **9**, 591–597 (1981)
13. Li, F., Zhang, B., Liu, B.: Ternary weight networks. In: NIPS, Workshop on EMDNN (2016)
14. Lichman, M.: UCI Machine Learning Repository (2013)
15. Lin, Z., Courbariaux, M., Memisevic, R., Bengio, Y.: Neural networks with few multiplications. CoRR, abs/1510.03009 (2015)
16. Moreno, R., Metnitz, P., Almeida, E., Jordan, B., Bauer, P., Abizanda, R., Campos, R.A., Iapichino, G., Edbrooke, D., Capuzzo, M., Le Gall, J.-R.: SAPS 3-from evaluation of the patient to evaluation of the intensive care unit. Part 2: development of a prognostic model for hospital mortality at ICU admission. Intensive Care Med. **31**, 1345–1355 (2005)
17. Nguyen, T.D., Phung, D., Huynh, V., Lee, T.: Supervised restricted Boltzmann machines. In: UAI (2017)
18. Peters, A., Hothorn, T., Lausen, B.: ipred: Improved predictors. R News **2**(2), 33–36 (2002)
19. Salakhutdinov, R., Hinton, G.: Deep Boltzmann machines. In: AISTATS (2009)
20. Salakhutdinov, R., Hinton, G.: A better way to pretrain deep Boltzmann machines. In: NIPS (2012)
21. Salakhutdinov, R., Hinton, G.: An efficient learning procedure for deep Boltzmann machines. Neural Comput. **24**(8), 1967–2006 (2012)
22. Salakhutdinov, R., Larochelle, H.: Efficient learning of deep Boltzmann machines. In: AISTATS (2010)
23. Srivastava, N., Salakhutdinov, R.: Multimodal learning with deep Boltzmann machines. J. Mach. Learn. Res. **15**, 1929–1958 (2014)
24. Srivastava, N., Salakhutdinov, R., Hinton, G.: Modeling documents with deep Boltzmann machines. In: UAI (2013)
25. Still, C.D., et al.: A probability score for preoperative prediction of type 2 diabetes remission following RYGB surgery. Lancet Diabetes Endocrinol. **2**(1), 38–45 (2014)
26. Ustun, B., Rudin, C.: Supersparse linear integer models for optimized medical scoring systems. Mach. Learn. **102**, 349–391 (2015)
27. Ustun, B., Rudin, C.: Learning optimized risk scores from large-scale datasets. In: KDD (2017)
28. Zhang, Y., Salakhutdinov, R., Chang, H.-A., Glass, J.: Resource configurable spoken query detection using deep Boltzmann machines. In: ICASSP (2012)
29. Zhu, C., Han, S., Mao, H., Dally, W.J.: Trained ternary quantization. In: ICLR (2017)

A New Approach for Tuned Clustering Analysis

Roni Ben Ishay[1(✉)], Maya Herman[1], and Chaim Yosefy[2]

[1] The Open University of Israel, Raanana, Israel
ronibenish@gmail.com, maya@openu.ac.il
[2] The Barzili Medical Center Campus, Ben-Gurion University, Ashkelon, Israel
chaimy@bmc.gov.il

Abstract. In this work, we present a new data mining (DM) approach (called *tuned clustering analysis*), which integrates clustering, and tuned clustering analysis. Usually, clusters which contain borderline results may be dismissed or ignored during the analysis stage. As a result, hidden insights that may be represented by these clusters, may not be revealed. This may harm the overall DM quality and especially, important hidden insights may be uncovered. Our new approach offers an iterative process which assist the data miner to make appropriate analysis decisions, and avoid dismissing possible insights. The idea is to apply an iterative DM process: clustering, analyzing, presenting new insights, or tuning and re-clustering those clusters which have borderline values. Clusters with borderline values are chosen and a new sub-database is built. Then, the sub-database is split, based on the attribute with the highest Entropy value. The tuning iterations, continues until new insights were found, or if the clusters quality are below a certain threshold. We demonstrated the *tuned clustering analysis* on real Echo heart measurements, using *km-Impute* clustering algorithm. During the implementation, initial clusters were produced. Although the quality of the clusters was high, no new medical insights were revealed. Therefore, we applied a clustering tuning and succeeded in finding new medical insights such as the influence of gender and the age on cardiac functioning and clinical modifications, with regard to resilience to diastolic disorder. Applying our approach has successfully managed to reveal new medical insights which were restored from borderline value clusters. This stands in contrast to traditional analysis methods, in which these potential insights may be missed or ignored.

Keywords: Data mining · Clustering · Clustering analysis · Imputation
Missing values · Medical data mining

1 Introduction

The use of Data Mining (DM) algorithms and techniques such as Clustering Analysis, is very common in recent years [1]. The aim of DM is to reveal new knowledge and insights by applying DM on a very large dataset. Clustering Analysis applies a

The original version of this chapter was revised: The spelling of the third author's name was corrected. The correction to this chapter is available at https://doi.org/10.1007/978-3-319-96136-1_35

P. Perner (Ed.): MLDM 2018, LNAI 10934, pp. 436–452, 2018.
https://doi.org/10.1007/978-3-319-96136-1_34

clustering algorithm on a dataset which builds group of clusters [2]. The members of each cluster are relatively similar, while members from different clusters are distinct from each other. Basically, clustering process consists of the following stages: (1) Data collection and pre-processing (2) Clustering (3) analyzing the results and revealing new insights [1]. The analysis stage is challenging and most important, as it determines the success of the entire DM. The analysis of the clusters is comprised of quality measurements, and content assessments. Cluster quality measurement calculates the amount of similarity within clusters, and the dis-similarity between different clusters. The greater the similarity within clusters as well as the dis-similarity between different clusters the higher the cluster quality. The content assessment is done by a content expert. Based on the subject of interest, the expert examines the values of centroids which represent the clusters. From the expert's perspective, there might be three types of centroids: Type 1– centroids which clearly reveal new insights. Type 2 – centroids which definitely do not lead to new knowledge, and Type 3 – centroids which contain borderline or inconclusive values. Often, centroids of Type 3 are discarded by the expert as non-valuable clusters. We claim, that despite the inconclusiveness of Type 3 centroids, it may contain hidden information. Therefore, further processing and investigation of Type 3 clusters may reveal new insights. In this work we present a new approach for the cluster analysis the clusters process. This new approach suggests that the investigators explore potential hidden insights in borderline clusters, which often are ignored. This approach is a continuation of our previous work in which we developed a novel clustering algorithm (*km-Impute*) in order to integrate imputation of missing values and clustering [3].

The *km-Impute* algorithm was tested on a dataset of wine quality measurements which was restored from the UCI repository [4]. During the analysis process, we noticed that although the quality of clusters was fine, revealing new insights was not straightforward, especially with regard to Type 3 clusters. Discarding of Type 3 clusters may cause a negative effect on the DM results. To minimize the discard of borderline clusters, in this work we offer a new approach which is a framework. It consists of the following iterative steps: clustering, quality evaluation, content assessment, and sub-clustering. The process continues until new insights are found, or the quality of the clusters is insufficient (according to pre-defined measures). This new approach provides guidelines and benchmarks, which helps to improve usefulness of the cluster analysis process. In Sect. 2, we describe related studies from the literature. Section 3 describes the new approach for the clustering analysis process. Section 4 provides details of the experiment of the tuned clustering analysis on real Echo-heart measurements. In Sect. 5 we present the results. We discuss the results in Sect. 6. Finally in Sect. 7 we conclude this work, and suggest ideas for further research.

2 Related Work

In every DM process, the mining results are analyzed based on quality measures. In the literature several DM techniques are mentioned and in each one DM analysis is done [1]. In this work we focus on clustering. Therefore, we will describe methods aimed at assessing the clustering result on various domains. In addition, techniques for the assessment of data mining on Echo heart data will be reviewed. An evaluation measure for stream clustering called Cluster Mapping Measure (CMM) is presented in [5]. CMM indicates different types of errors by taking the important properties of data streams into account. Na et al. [6] describe the concept of a hybrid strategy for clustering validity. It is based on the following two measures: the first one – analysis - analyzes the goodness of each partition, and the second measure enables selection of the best partition between those having good but very close values of the first measure [6]. A framework for hierarchical clustering algorithm evaluation in the domain of functional genomics, is suggested by Guo [7]. First construction of a Bayesian network model that simulates the desired characteristics of the domain is applied. Then, the clustering algorithm is run on sample from the network data. Finally, the performance of the algorithm evaluated based on how well it partitions different sub-networks in the model [7]. Tsipouras et al. [8] suggest an automated diagnosis of coronary artery disease, based on Fuzzy modeling. The idea is to apply four-stage methodology using an invasive cardiology dataset. First, a decision tree is inducted. Then a crisp rule-based classifier is built. Finally (stages 3 and 4), the fuzzy model is deployed and optimized. The results are evaluated by a decision support system. Other methods apply supervised machine learning algorithms such as Naïve Bayes, k-nearest neighbors (k-NN), Decision Tree and Classification based on clustering in order to predict heart diseases [9–11]. A prototype of an Intelligent Heart Disease Prediction System (IHDPS) is described in [10]. The results show that each technique has its unique strength in realizing the objectives of the defined mining goals. IHDPS can answer complex questions such as "what if" queries. Using medical profiles such as age, gender, blood pressure and blood sugar it can predict the likelihood of patients getting heart disease [10]. A genetic algorithm is used to determine the attributes which contribute more towards the diagnosis of heart ailments which indirectly reduces the number of tests which are needed to be taken by a patient [12]. Another approach is to integrate clustering and statistical analysis for supporting cardiovascular disease diagnosis [13]. It considers combining statistical inference with clustering in the preprocessing phase of data analysis.

3 The Approach

3.1 The Concept

Our goal is to address the following issues which arise during clustering analysis:

1. Are the cluster quality measures such as density and separation satisfactory? When should we consider tuning the clustering?
2. If tuning of clusters is required:
   - What clustering parameters should be modified for the sub-clustering?
   - How should partial DB be selected for sub-clustering.
   - What should the criteria be to terminate the tuning's iterations?

Suppose a clustering algorithm (such as *km-Impute*) is applied on a DB. The clusters are examined in two aspects: First, their quality is calculated (see 3.3) and a content expert makes a content assessment - trying to reveal new insights from the clusters.

In this work we focus on the cases where clusters contain borderline or inconclusive values. In this case, we suggest tuning the clusters by iterative clustering (sub-clustering) in order to improve the probability of revealing new insights.

The *tuned clustering analysis* starts by applying the *km-Impute* algorithm (Fig. 1 step 1). It builds clusters while imputing missing values. The quality of clusters is calculated based on quality measures (step 2). If the quality of the clusters is not sufficient, and the number of iteration attempts is smaller than the maximum allowed, then (Step5):

- Increase the size of the core sample DB of *km-Impute* (step 6).
- Increase the number of clusters (step 6).
- Start a new iteration (step 1).

If the quality of the clusters is sufficient, and the content expert has found new insights (step 3), publish the new insights and stop (step 4).

If no insights are found (step 3):

- Choose clusters with borderline values, to be used for sub-clustering, (step 7).
- Calculate the highest entropy attribute (EA) (step 8).
- Split the chosen clusters to sub databases, according to the EA (step 9).
- Start a new iteration (step 1).

If the number of iterations has reached the maximum (step 5) but no new insights were revealed – stop without new insights.

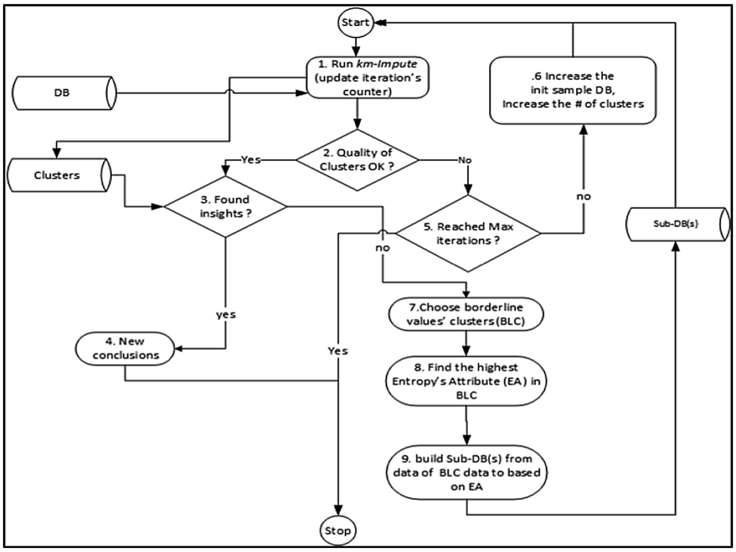

Start
1. Run km-Impute (update iteration's counter)
DB
.6 Increase the init sample DB, Increase the # of clusters
Clusters
Yes
2. Quality of Clusters OK ?
No
3. Found insights ?
no
5. Reached Max iterations ?
no
Sub-DB(s)
yes
7.Choose borderline values' clusters(BLC)
4. New conclusions
Yes
8. Find the highest Entropy's Attribute (EA) in BLC
9. build Sub-DB(s) from data of BLC data to based on EA
Stop

Fig. 1. The flowchart of the new approach for the tuned clustering analysis.

3.2 The Pseudo Code of the *Tuned Clustering Analysis*

Notations

- DB – the database to be processed and analyzed.
- insights – a summary of new insights which were revealed from the DM.
- *km-Impute* – the clustering algorithm.
- Clusters – the set of clusters that are produced by *km-Impute*.
- calculateQuality – calculate the quality of clusters (by Silhouette Coefficient [14]).
- Quality – store the quality of clusters.
- chooseBorderlineClusters – return data points from borderline values' clusters.
- BLC – data from borderline values' clusters.
- EA – the attribute with the highest Entropy value.
- SplitDB – build a sub DB based on the EA.
- findHighestEntropyValueAttribute – return the highest Entropy value attribute.
- qualityThreshold –pre-defined the minimum required quality of clusters (optional).
- contentAssessment – content assessment of the clusters (by the content expert).
- newInsights – return true new insights were found.
- k – the number of clusters.
- initSample Db Size – the size of the core sample DB which initiate the clustering.
- changeClusteringParmeters – change the size of sample DB, or the number of clusters.
- iterationsCounter – the clustering attempts' counter.
- maxIterations – maximum iterations for clustering threshold.

Pseudo Code

```
Program Tuned Clustering analysis
   1.  Input: DB
   2.  Output: insights
   3. Repeat
   4.      Clusters = km-Impute(DB)
   5.      Quality = calculateQuality(clusters)
   6.    if(Quality >= qualityThreshold) then //optional
   7.       newInsights=contentAssessment(clusters)
   8.       if(newInsights==true) then
   9.          Output(insights)
   10.      else
   11.        BLC=chooseBorderlineClusters(Clusters)
   12.        EA= findHighestEntropyValueAttribute(BLC)
   13.        DB=SplitDB(BLC, EA)
   14.   else // quality is not sufficient
   15.      changeClusteringParmeters(initSampleDbSize,k)
   16.     iterationsCounter++
   17. Until (newInsights) or
             (iterationsCounter ==maxIterations)
end Program
```

Pseudo Code Description
Line 3: build the DB to be clustered.
Lines 3–17: the main iterative loop.
Line 4: Apply the *km-Impute* clustering algorithm. It builds clusters while imputing missing values, and return a group of clusters (Clusters).
Line 5: calculate the quality of clusters, using the Silhouette Coefficient.
The quality of clusters is calculated as the average of the Silhouette Coefficient (SilC) of all clusters [14] (step 2). SilC is a factor of the tidiness in-between clusters and the distance between different clusters. The SilC of each item is calculated as follows: $s_i = (b_i - a_i)/\max(a_i, b_i)$.
Where *i* is a data point in a cluster, a_i is the average distance of *i* with all other data points within the same cluster, b_i is the lowest average distance of *i* to all data points in any other cluster, of which *i* is not a member of, s_i is the Silhouette of data point *i*. The range of SilC is between −1 to 1. The closest the SilC to 1, the better the quality of the cluster is.
The total quality of clusters is stored in Quality.
Lines 6, 7: if the clusters' quality is above threshold (optional), then make a content assessment.
Line 8, 9: if new insights were found, set newInsights flag to true, and output the insights.
Line 10–13: else – (quality is below the threshold) find borderline clusters (BLC), find the highest Entropy attribute value (EA) and build subDB(s).
Line 14–16: no new insights were found. Change the *ta*parameters: Increase the size of the initial DB sample, and the number of clusters to produce.
Line 17: termination criteria options:

- New insights were found.
- Reached the maximum allowed iterations.
  - Otherwise – start a new iteration.

Complexity Analysis
The method is built of one iterative repeat-until loop. Algorithm *km-Impute* runs with a complexity of $O(N^{Dk+1}\log N)$, (where N is the DB size, K is the number of clusters and D is the number of attributes). Let M be the number of iteration. Therefore the total number of actions is $M * O(N^{Dk+1}\log N)$. Assuming that $M << N$, the total complexity will be $O(N^{Dk+1}\log N)$.

4 Experiments

4.1 Objectives

The objectives are medical and DM objectives. The main medical objectives are to identify quantitative Echo heart measurements, which may indicate the presence of the future possibility of heart diseases. While the DM objective are to find out whether:

1. Clusters with pure quality may insure the revealing of new insights.
2. Tuning the clusters (sub-clustering) of clusters with inconclusive values, may reveal new insights.

4.2 Medical Overview

In the medical literature several lines of research explore the influence of age and gender on the cardiac function [15–17]. As age rises, the probability of suffering from heart disease rises accordingly. Above 65 years old, people are at a higher risk of getting a heart attack, with 1 in 5 patients who get a heart attack likely to die during the following 10 years (see Framingham score [1]). In general, during the range of age of 46–64 men have a higher tendency to develop heart disease than women do. Research shows that prior to their menopause, women are relatively protected from heart disease, while men may get heart disease at a younger age [15, 18]. As women go through menopause, the risk of developing heart disease gets significantly higher. Between the ages of 46 to 64, more men may die as a result of heart disease complications compared to women (39% more) [18]. After 64 years old, the trend changes and more women are likely to die from a heart attack (22%). The left ventricle in both men and women becomes somewhat thicker as they get older [19]. As a result, the left ventricle's doorway becomes narrower, and the systolic blood pressure rises. The increase in the thickness of the left ventricle, combined with shortening of its length, causes a modification of the left ventricle's shape to an elliptic form. As people age, the probability of Aortic Regurgitation rises, which may influence about 16% of the elderly population for both genders [19].

4.3 The Echo Heart Measurements Database

For the DM process, we used Echo heart measurements database (Echo-DB). It contains about 26,000 records, which were measured during the years 1997 to 2006 (Table 1). The Echo-DB is not complete and has some missing values. Therefore, an imputation of missing values is needed.

Table 1. Echo heart measures type and range. (*A-Area, Diam-Diameter)

Attribute	Measure	Type	Range (min… max)
Age		Int	(39…95)
Gender	Male/Female	Int	(0, 1)
Aortic Root Diameter (Ao_Diam)	mm	Float	(20…36)
Ascending Aorta Diameter (AsAo_Diam)	mm	Float	(20…36)
Left Atrial Diameter (LA_Diam)	mm	Float	(20…40)
Left Atrial Area (LA_Area)	cm^2	Float	(13…17)
Left Ventricle End Diastolic Diameter (LVEDD)	mm	Float	(35…54)
Left Ventricle End Systolic Diameter (LVESD)	mm	Float	(23…40)
Left Ventricle Ejection Fraction (LV_EF)	%	Float	(55…70)
Left Ventricle Septal Thickness (LV_STH)	mm	Float	(7…11)
Left Ventricle Posterior Wall thickness (LV_PWTH)	mm	Float	(7…11)
Right Ventricle Diameter (RV_Diam)	mm	Float	(30…40)
Right Atrial Area (RA_Area)	cm^2	Float	(13…17)

4.4 The Implementation

The main goal of the implementation was to demonstrate the *Tuned Clustering Approach* on a real database of Echo heart measurements. A secondary goal was to find new insights with regard to our medical issue of interest: find out whether gender and the age of a patient influence his cardiac function and clinical modifications, with regard to his resilience to diastolic disorder. The implementation is applied as follows: The *km-impute algorithm* was run and six clusters were recognized, and their quality was checked. Since their quality was high, a medical expert made a content assessment, but did not reveal new medical insights. Therefore, fine-tuning was required. We selected particular clusters for sub-clustering, based on a root attribute with the highest entropy value. The quality of the sub-clustering was checked and a content assessment was made.

5 Results

We ran the *km-Impute* (Fig. 1 step 1) to get the maximum SilC. The quality of clusters was assessed by the Silhouette Coefficient (SilC) measure [20] (Fig. 1 step 3). The silhouette is a measure of the similarity of data points within cluster (cohesion) compared to other clusters (separation). The silhouette range is −1 to +1. A value closed to +1 indicates that the data points within cluster are strongly similar, while poorly similar to neighboring clusters. After 7 runs, run #5 achieved the best SilC average (Table 2).

Table 2. Silhouete Coefficient (SilC) avarage on Echo-DB

Run#	SilC average
1	0.11
2	0.09
3	−0.07
4	0.01
5	0.21
6	0.02
7	0.13

The *km-Impute* produced 6 clusters (Table 3, and Fig. 2) which were assessed by a medical expert (Fig. 1 step 3). Clusters 1, 2, 3, 4 (Table 3) contain abnormal values of LA-Diam, LA-Area, LVESD, LVESD, Est-LVEF and RA-Area, which indicate various degrees of cardiac disease. These values are borderline and may hide new medical information, therefore they may be medically interesting. Clusters 5, 6 contains normal cardiac values, and do not contain any new medical insights.

Table 3. The results of run #5. Clusters of patients at age range 39–95.

Cluster	RA-Area	R-Ventricle	Post-Wall	IV-Septum	Est-LVEF	LVESD	LVEDD	LA-Area	LA-Diam	Ascending-Ao	Aortic-root
1	16.50	30.71	10.36	10.93	59.81	32.3	49.4	20.20	37.71	34.18	33.64
2	15.77	30.42	9.98	10.60	61.52	30.0	47.0	20.57	37.41	33.75	30.34
3	18.11	31.45	10.14	10.80	34.39	45.6	56.6	23.73	41.25	34.72	34.25
4	17.07	31.21	10.19	10.86	37.39	40.4	52.2	23.97	40.73	34.44	31.11
5	13.54	30.01	8.68	8.90	64.27	30.2	48.7	15.20	33.11	31.16	30.41
6	12.36	29.68	8.06	8.26	64.79	28.2	47.1	14.32	31.49	30.86	27.46

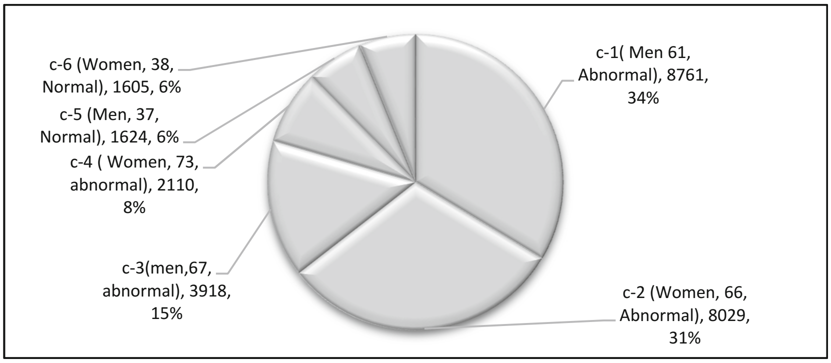


Fig. 2. The distribution of clusters, produced by clustering the entire Echo heart measurements. (Age range: 39–95)

The results were validated by the “null hypothesis” [21] test as follows: The gender and age of the patient do not influence his cardiac function and clinical modifications, with regard to his resilience to diastolic disorder. It was tested by the Cochran-Mantel-Haenszel test [21] using the open source tool Rstudio [22]. The test shows that the null hypothesis is denied, with a significance level of p-value < $2.2e^{-16}$. Therefore, we conclude that the gender and the age of the patient, do influence his cardiac function and clinical modifications, with regard to his resilience to diastolic disorder.

Since DM process did not reveal new knowledge, tuning the clusters (sub-clustering) was required. We picked data points of clusters 1, 2, 3, 4 (which contained borderline values) (Fig. 1 step 8), which were used to build a sub-DB. Then we calculated the attributes’ Entropy, and found that Age has the highest Entropy value. The sub-DB split to three datasets by Age Range as follows: 1. Patients aged 50–60. 2. Patients aged 61–70. 3. Patients aged 71–95. Later, on each dataset we ran the *km-Impute*, in order to find possible hidden information on the selected data. The run that achieved the best SilcC result was chosen (Fig. 1 step 3): Group-1 run #4, group-2, and run #4 and for group-3 run #5 (Table 4).

Table 4. SilC average for 3 group of patients.

Run#	SilC average group 1 (50–60)	SilC average group 2 (60–71)	SilC average group 3 (71–95)
1	−0.01	−0.11	−0.07
2	−0.16	0.09	0.08
3	−0.08	−0.02	0.01
4	0.14	0.14	−0.01
5	0.02	−0.10	0.10

In Group-1, the clusters 5.6 (age range 50–60) contained normal heart measures hence are not interesting medically (Fig. 3). Clusters 1, 2 represent men aged 53–58. Their Est-LVEF is below normal. Their systolic function is borderline with a compensatory enlargement of their left ventricle. The reason is high probability of diastolic disorder, which matches the patient's age. Clusters 3, 4 refer to women aged 53–58 with LA-Area above normal. These women (similar to men in clusters 1, 2) also suffer from enlarged left ventricle, although with a smaller left atrium (women = 48 cm^2, men = 51 cm^2). Note that women's hearts function better than men's. Since the women's heart volume is smaller, they respond sooner to diastolic disorder by enlarging their left ventricle. In addition, the values of their ejection fraction are higher than men (57%–50% vs. 51%–52% accordingly). In men, in comparison, the heart is larger and its functionality is borderline (clusters 1, 2). The LA-Area of men is 20 cm^2–21 cm^2, while in women the range is 19 cm^2–20 cm^2. More men are diagnosed with abnormal heart function than women: 2,594 and 1,580 accordingly.

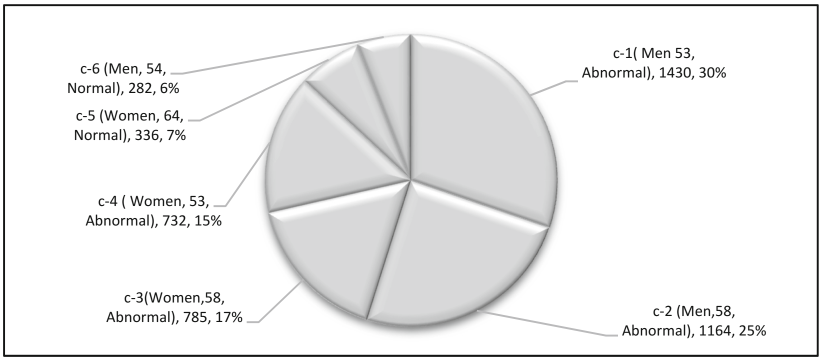


Fig. 3. The distribution of clusters produced by clustering the age range of 50–60 (cluster of group-1) Echo heart measurements.

The results of group-2 (age range 61–70), all clusters contain abnormal measures (Table 5, and Fig. 4). Clusters 1, 2 refer to women aged 63–68 with LA-Area above normal. Clusters 3, 4 represent men aged 63–68 with IV-Septum and LA-Area above normal. Clusters 5, 6 refer to men aged 64–69 were LVESD, Est-LVEF, LA-Area, LVEDD and RA-Area are all above normal. Clusters 1–4 represent both women and men with relatively good heart function. These clusters are similar to clusters 1, 2, 3, 4 of group-1 (patients aged 50–60). Note that men's hearts are larger than women's: LVEDD = 49 mm (clusters 3, 4) and LEVDD = 48 mm (clusters 1, 2) accordingly. Clusters 5, 6 refer to men with low systolic functioning (moderate to severe) where Est-LVEF is ranged 32%–33%, enlarged Left Ventricle (LA-Area 32 cm^2 to 55 cm^2) and borderline Right Ventricle diameter ($\sim$18 mm). These measures indicate high pressure in the left atrium which causes the enlargement of the right ventricle. At the age of 61–70 there are more men than women with abnormal or borderline heart condition: 3,800 versus 3,140 women.

Table 5. Clusters of patients at age range: 61–70.

Cluster	R-Ventricle	Post-Wall	IV-Septum	Est-LVEF	LVESD	LVEDD	LA-Area	LA-Diam	Ascending-Ao	Aortic-root	RA-Area
1	16.33	30.61	10.23	10.8	56.46	32.69	48.8	21.6	38.85	34.32	30.68
2	15.53	30.38	9.96	10.4	57.67	32.15	48.7	20.1	37.84	34.18	30.62
3	16.67	30.77	10.70	**11.3**	59.12	32.47	49.4	20.5	38.48	35.06	34.20
4	16.13	30.46	10.51	**11.1**	59.72	32.43	49.7	19.7	38.14	34.59	33.99
5	**18.13**	31.20	10.11	10.8	**33.76**	**46.16**	**57.1**	23.7	**41.43**	34.41	34.13
6	**18.27**	31.43	10.17	10.7	**32.93**	**46.96**	**57.7**	23.6	**41.33**	35.07	34.44

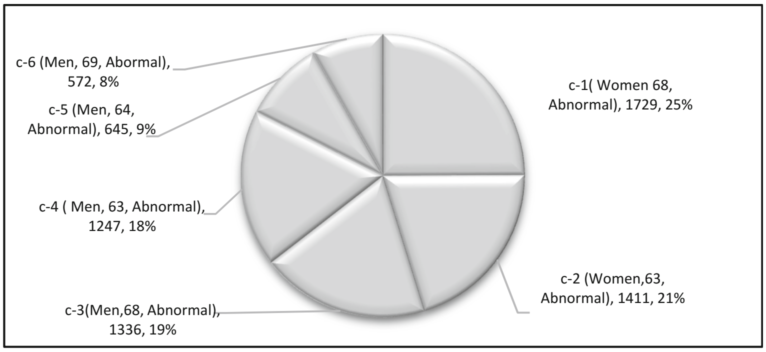


Fig. 4. The distribution of clusters produced by clustering the age range of: 61–70 (group-2) Echo heart measurements (bold measurements are abnormal values).

Group-3 (ages 71–95) also contain above normal measures (Table 6, Fig. 5). Clusters 1 and 4 refer to men aged 75 and 84 respectively with exceptional LA-Area, IV-Septum, LA-Diam and RA-Area measures. Cluster 5 represents women at age 87 with exceptional LA-Area, IV-Septum, Est-LVEF and RA-Area. Cluster 6 refers to women with exceptional LA-Area, LA-Diam, LVESD and Est-LVEF. At this range of ages, we found that the left atrium of women is relatively normal (clusters 2, 3). Men on the other hand, suffer from abnormal left atrium function, but the women's heart muscle is less enlarged in comparison to men.

Table 6. Clusters produced for patients at age range: 71–95.

Cluster	RA-Area	R-Ventricle	Post-Wall	IV-Septum	Est-LVEF	LVESD	LVEDD	LA-Area	LA-Diam	Ascending-Ao	Aortic-root
1	**17.92**	31.10	10.62	**11.30**	**49.04**	38.08	52.47	**22.90**	**40.35**	35.55	34.61
2	16.34	30.50	10.43	**11.18**	60.0	30.64	47.30	**21.50**	38.42	34.79	30.99
3	16.63	30.72	10.57	**11.43**	60.52	29.89	46.34	**22.49**	39.14	34.79	31.00
4	**18.37**	31.58	10.75	**11.56**	**52.37**	35.51	50.28	**23.42**	**40.29**	35.96	34.48
5	**17.14**	30.83	10.72	**11.71**	**54.63**	31.23	45.81	**23.57**	39.73	35.04	31.04
6	16.83	31.07	10.11	10.87	**33.50**	**42.13**	53.00	**24.52**	**41.10**	34.72	31.27

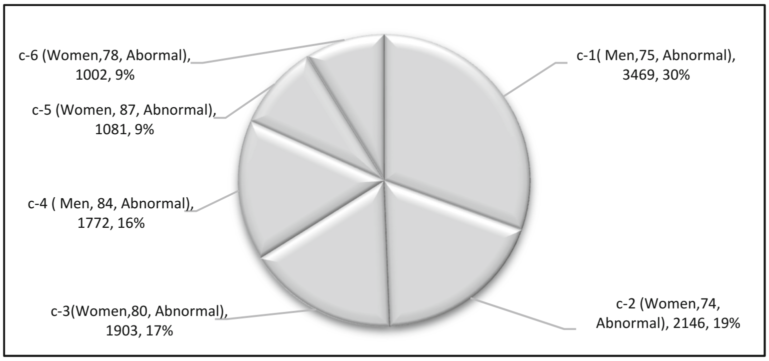


Fig. 5. The distribution of clusters produced by clustering the age range: 71–95 (group-3) Echo heart measurements.

6 Discussion

Our medical issue of interest is to find whether gender and age of the patient influence his cardiac function and clinical modifications. Regarding his resistance to diastolic disorder, we compared the clusters, produced in each period. One of the most significant measures used to examine the heart function is the left atrium area (LA-Area). In Figs. 6 and 7, we can see the range of LA-Area values of women and men, at three age decades. For both women and men, the LA-Area increases linearly with progress of age, as can be see through the dotted lines in Figs. 6 and 7.

The LA-Area values of women at the age range of 50–60 is 14.1 cm^2–19.67 cm^2 (Fig. 6) In comparison, men have slightly larger LA-Area (15.8 cm^2 to 20.46 cm^2) (Fig. 7). This indicates that the difference between women and men starts already at the age range of 50–60, where women's hearts function better than in men. At the age range of 60–71 the LA-Area of both genders becomes larger: women with LA-Area of 20.11 cm^2 to 21.63 cm^2 and in men it ranges from 19.72 cm^2 to 23.64 cm^2. We noted that the enlargement of the LA-Area in men happens faster than in women. Also, the variance between men in the same group is higher than in women. These results indicate that at the age range 60–71, the heart function of women is more homogenous than men at the same age range. The enlargement of the LA-Area reflects the left ventricular diastolic pressure. This pressure is responsible for the filling capacity of the left ventricle. The capacity is a major factor with regard to the production of cardiac output. As the left ventricular diastolic pressure becomes lower, the filling capacity volume of the left ventricle increases faster. This leads to the improvement of the patient's exercise capacity. On the other hand, high diastolic pressure will lead to less left ventricle filling capacity. As a result, the backpressure on the lung vasculature will increase and may cause an effort dyspnea. The effects on a patient are difficulties or inabilities during physical efforts. Revealing the group of patients who may suffer from inability to accommodate heart filling during some physical effort is very important. This may enable applying a better follow up plan for patients, and shorten the time between periodic examinations. In addition, it may be used as a guideline for the medical staff, to decide about the optimal time to start pharmacological treatment (reducing LVEDD pressure and improving Left ventricular compliance). Additionally, the differences of the behavior of the left ventricular between women and men over three decades (Figs. 6 and 7) is significant: These differences may provide valuable information about the potential of patients to develop heart disease. The new medical treatment approach is moving toward patient-oriented treatment. By carefully considering these differences, and by providing each individual with tailored therapy and treatment, we may improve future prognosis.

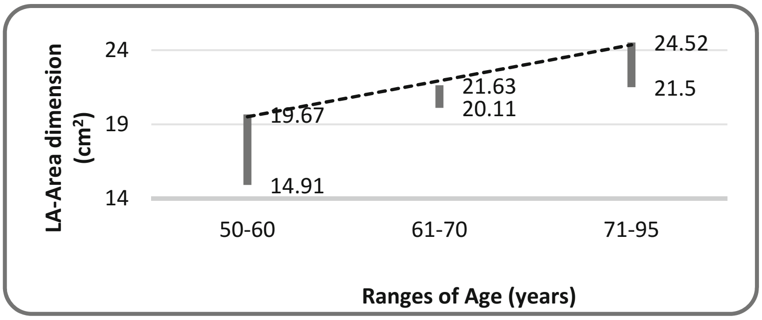


Fig. 6. Left atrium area of women in 3 age range periods.

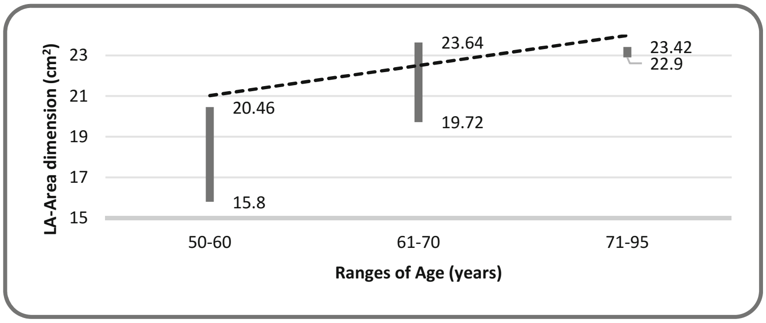


Fig. 7. Left atrium area of men in 3 age range periods.

In men, we found interesting phenomenon, regarding their heart function; men may be in one of two groups: The resilient - men with relatively good heart function and a small LA-Area (19.72 cm^2). The sick - men with heart disease, who suffer from rapid deterioration in comparison to their condition at age 50–60. These men have a very large LA-Area (23.64 cm^2) – which is wider than women at the same age range (21.63 cm^2). At age 71–95, the left atrium area of both women and men have become similarly wider: between 21.5 cm^2 to 24.52 cm^2 in women (Fig. 6) and 22.9 cm^2 to 23.42 cm^2 at men (Fig. 7**).** Note that the group of sick men at this age is larger than the women's group. The comparison of the average Left Atrium Area values between women and men (Fig. 8) shows that men have a wider left atrium at all ages. As they age, the difference between the left atrium of women and men gradually gets smaller, until at age 72–95 when the area is almost the same. This indicates that up to the age of 61–70, men are less resilient to heart disease than women **–** hence their left atrium area is wider. Notice that some men aged 61–70 have managed to function relatively well even with borderline heart measures. In contrast, another group could not function normally, and they have suffered from a rapid deterioration in their heart condition. At age range 71–95, their heart function was very bad.

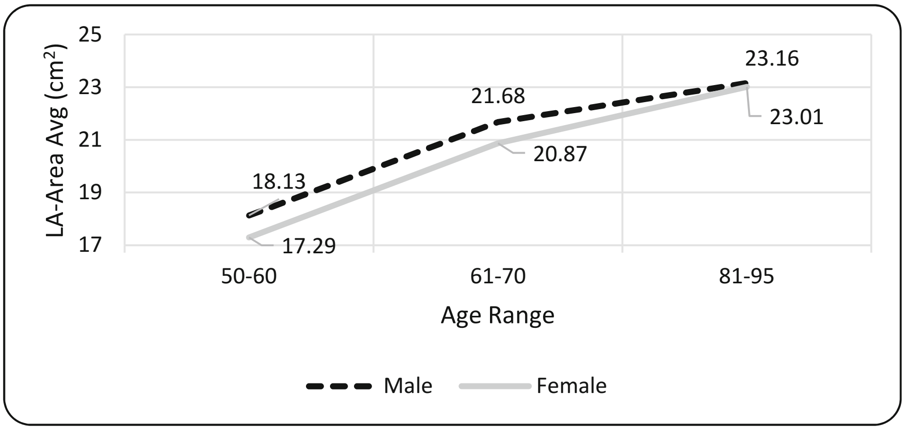


Fig. 8. The comparison of Left atrium area between women and men 3 age range periods.

Comparison between clusters of men of ages 50–60 to clusters of men of ages 61–70 indicates on interesting medical insight. Men of ages 50–70 are divided into 2 groups with respect to their resilience to heat disease. The first group: Men in their 50–60's (Fig. 3, clusters 1, 2) had a good functioning heart although borderline, but as they became older (61–70) (Table 5, clusters 3, 4, 5, 6) they suffered from very fast deterioration which occurs relatively soon. The second group: Men in the age ranges of 50–60 and 61–70, whose heart was functioning adequately, although their heart measures were borderline. These men were relatively durable to heart disease. At the age range 50–60, most of the men have borderline heart measures (2,594) where the rest are durable (282). At age range 61–70, the sick/borderline group of men is smaller than the durable ones (1,217 and 2,583) respectively.

The results show that the resilience of women to heart diseases is better than that of men. At age range, 50–60 the response of women to diastolic disorder is self-protection by the enlargement of the left atrium. During the next periods, 61–70 and 71–95, women's hearts become thicker gradually. In contrast, men are divided into two groups: The resilient: men that have managed to function well along the period of 50 to 95. The sick: men that have suffered from relatively early heart tiredness. During the enlargement of their left atrium area at the age of 50–60, their heart function has deteriorated gradually. In addition, we found that while men at 60–71 had relatively good or moderate heart function, at age 71–95 their heart failure was significantly more severe than women. Note that these findings were not revealed when clustering the entire Echo-DB (Table 5); further tuning was required. The clinical implication of the tuning of clusters and the revealing of new insights, leads to the suggestion that a group-specific medical management approach may be indicated. The group of women and the group of the resilient men (aged 60–71) should be treated according to the regular protocol. In contrast, the group of sick men should be monitored earlier, in order to control their heart condition and to prevent future degradation. This management may include: pharmacological or non-pharmacological treatment, specific diet, minimization of salt consumption, weight control, physical and sport activities. These may improve the future prognosis of the patients.

7 Conclusion and Further Research

In this work, we have presented a new method for analyzing and treating DM results. The new approach - *a Tuned Clustering Analysis* defines steps and guidelines to assist the data miner during the DM process such as: evaluating the quality of clusters, involvement of the content expert and his assessments, and applying clustering tuning. The method is iterative, comprising: Clustering, quality and content evaluating, and cluster tuning (sub-clustering). The tuning will be applied if no new insights were revealed from the primary clustering. Our approach improves the probability of the DM process, to reveal new insights even from clusters with borderline values. The *Tuned Clustering Analysis* was applied on a real Echo heart measurements. The clustering was implemented using our *km-Impute* clustering algorithm. Two iterations applied. The first iteration did not reveal new insights. After the second iteration (tuning of clusters) we successfully managed to reveal new insights. With respect to medical insights, interesting insights were revealed with regard to the diagnosis of Echo heart measurements databases: We found that the heart conditions of all women gradually retrogrades as ageing progresses. The group of women who start to suffer from severity of heart diseases as a function of their age, is smaller than the group of men. Despite this, we identified two sub groups of men. The first sub group: men who suffer from heart diseases which become more severe as they age. The second sub group: men who have immunity relatively to heart diseases. According to these findings, men who belong to the first group should be monitored earlier to control their potential degradation.

Regarding DM, further research ideas would be implementing this new method on other clustering algorithms such as neural networks. The new medical insights may help the diagnosis and treatment of heart diseases in men.

References

1. Han, J., Pei, J., Kamber, M.: Data Mining: Concepts and Techniques. Elsevier, New York (2011)
2. Srinivas, K., Rani, B.K., Govrdhan, A.: Applications of data mining techniques in healthcare and prediction of heart attacks. Int. J. Comput. Sci. Eng. (IJCSE) **2**(02), 250–255 (2010)
3. Ben Ishay, R., Herman, M.: A novel algorithm for the integration of the imputation of missing values and clustering. In: Perner, P. (ed.) MLDM 2015. LNCS (LNAI), vol. 9166, pp. 115–129. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21024-7_8
4. Bache, K., Lichman, M.: UCI Machine Learning Repository (2013). http://archive.ics.uci.edu/ml. Accessed 1 May 2013
5. Kremer, H., et al.: An effective evaluation measure for clustering on evolving data streams. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 868–876. ACM, San Diego (2011)
6. Na, Y., et al.: HS-measure: a hybrid clustering validity measure to interpret road traffic data. In: Proceedings of the 5th International ICST Conference on Performance Evaluation Methodologies and Tools, pp. 274–280. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Paris (2011)
7. Guo, A.: A new framework for clustering algorithm evaluation in the domain of functional genomics. In: Proceedings of the 2004 ACM Symposium on Applied Computing, pp. 143–146. ACM, Nicosia (2004)
8. Tsipouras, M.G., et al.: Automated diagnosis of coronary artery disease based on data mining and fuzzy modeling. IEEE Trans. Inf. Technol. Biomed. **12**(4), 447–458 (2008)
9. Soni, J., et al.: Predictive data mining for medical diagnosis: an overview of heart disease prediction. Int. J. Comput. Appl. **17**(8), 43–48 (2011)
10. Palaniappan, S., Awang, R.: Intelligent heart disease prediction system using data mining techniques. In: IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2008. IEEE (2008)
11. Bhatla, N., Jyoti, K.: An analysis of heart disease prediction using different data mining techniques. Int. J. Eng. **1**(8), 1–4 (2012)
12. Anbarasi, M., Anupriya, E., Iyengar, N.: Enhanced prediction of heart disease with feature subset selection using genetic algorithm. Int. J. Eng. Sci. Technol. **2**(10), 5370–5376 (2010)
13. Wosiak, A., Zakrzewska, D.: On integrating clustering and statistical analysis for supporting cardiovascular disease diagnosis. In: 2015 Federated Conference on Computer Science and Information Systems (FedCSIS). IEEE (2015)
14. Zhang, S., Zhang, C., Yang, Q.: Data preparation for data mining. Appl. Artif. Intell. **17**(5–6), 375–381 (2003)
15. Chobanian, A.V., et al.: The seventh report of the joint national committee on prevention, detection, evaluation, and treatment of high blood pressure: the JNC 7 report. JAMA **289** (19), 2560–2571 (2003)
16. Zhao, R., et al.: Influences of age, gender, and circadian rhythm on deceleration capacity in subjects without evident heart diseases. Ann. Noninvasive Electrocardiol. **20**(2), 158–166 (2015)

17. Adams, K.F., et al.: Relation between gender, etiology and survival in patients with symptomatic heart failure. J. Am. Coll. Cardiol. **28**(7), 1781–1788 (1996)
18. Leinwand, L.A.: Gender is a potent modifier of the cardiovascular system. J. Clin. Invest. **112**(3), 302–307 (2003)
19. Karavidas, A., et al.: Aging and the cardiovascular system. Hell. J. Cardiol. **51**(5), 421–427 (2010)
20. Mirkin, B.: Clustering For Data Mining: A Data Recovery Approach (Chapman & Hall/CRC Computer Science). Chapman & Hall/CRC (2005)
21. Gandrud, C.: Reproducible research with R and R studio. Chapman and Hall/CRC (2016)
22. RStudio: An open source statistical language (2017). https://www.rstudio.com

Correction to: A New Approach for Tuned Clustering Analysis

Roni Ben Ishay, Maya Herman, and Chaim Yosefy

Correction to: Chapter "A New Approach for Tuned Clustering Analysis" in: P. Perner (Ed.): *Machine Learning and Data Mining in Pattern Recognition*, LNCS 10934, https://doi.org/10.1007/978-3-319-96136-1_34

The original version of this chapter contained an error in the third author's name. The spelling of Chaim Yosefy's name was incorrect in the header of the paper. The author name has been corrected.

The updated version of this chapter can be found at
https://doi.org/10.1007/978-3-319-96136-1_34

P. Perner (Ed.): MLDM 2018, LNAI 10934, p. C1, 2018.
https://doi.org/10.1007/978-3-319-96136-1_35

Author Index