The Pennsylvania State University

The Graduate School

MINING TEXTS AND SOCIAL USERS USING TIME SERIES AND LATENT TOPICS

A Dissertation in

Information Sciences and Technology

by

Tao Yang

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

May 2014

The dissertation of Tao Yang was reviewed and approved* by the following:

Dongwon Lee
Associate Professor of Information Sciences and Technology
Dissertation Advisor, Chair of Committee

Xiaolong (Luke) Zhang
Associate Professor of Information Sciences and Technology

Prasenjit Mitra
Associate Professor of Information Sciences and Technology

Bruce G. Lindsay
Professor of Statistics

Jim Jansen
Associate Professor of Information Sciences and Technology
Graduate Program Academic Coordinator

*Signatures are on file in the Graduate School.

Abstract

Knowledge discovery has received tremendous interests and fast developments in both *text mining* and *social user mining*. The main purpose is to search massive volumes of data for patterns as so-called knowledge. Knowledge can exist in different formats such as texts or numbers. Knowledge can be observed or hidden in different hierarchies. Knowledge can even be user-generated such as social content and social activity in Web 2.0 era. In this dissertation, we study a series of new knowledge discovery techniques using four data mining applications. First, we propose our novel framework on mining text databases using time series by bridging two seemly unrelated domains - alphabets strings and numerical signals. We study how various transformation methods affect the accuracy and performance of detecting near-duplicate texts in *record linkage*. Second, we develop new topic models on mining text documents using latent topics to tackle the noisy data problem in *document modeling*. We show how the incorporation of textual errors and topic dependency into the generative process affect the generalization performance of topic models. Third, we introduce our novel methods in *mining social content* using time series to classify user interests. We show the accuracy of our approach in both binary and multi-class classification of sports and political interests of social users. Finally, we introduce our generative modeling approach in *mining social activity* using latent topics to predict user attributes. We show the performance of our methods in predicting binary and multi-class demographical attributes of social users.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I am most grateful to my advisor, Dr. Dongwon Lee, for his guidance, encouragement, patience, and support during my Ph.D. study in College of Information Sciences and Technology at The Pennsylvania State University. He spent countless time and efforts guiding my research. This dissertation would not have been possible without his supervision and helps.

Second, I would like to thank my committee members, Dr. Xiaolong Zhang, Dr. Prasenjit Mitra, and Dr. Bruce Lindsay, for their valuable and insightful commentary on my Ph.D. work. I would like to give special thanks to Dr. Su Yan at IBM Almaden Research Center for providing useful suggestions and collaboration in part of my dissertation research.

Third, I would like to extend my thanks to my supervisors Dr. Vicki Williams and Dr. Bart Pursel in Teaching and Learning with Technology at Penn State, where I worked as a graduate assistant for two years. I would also like to thank my supervisors Dr. Vernon Chinchilli and Dr. Wenlei Liu in Department of Public Health Sciences at Milton S. Hershey Medical Center, where I spent two summers for my internships.

I want to acknowledge all my professors, colleagues, friends who are not mentioned above. Their encouragements and advice are really appreciated.

Last but not least, I would like to sincerely thank my parents, whose love and confidence in me has accompanied me in my life. Without their support, this dissertation would not have been possible.

Chapter 1

Introduction

1.1 Overview

In the real world, we are facing and need to deal with enormous data from various database applications, e.g., data warehouses, search engines, digital libraries, social networks and so on. As a result, data mining research focuses on methods, algorithms and tools to handle large amounts of data and has become an important part of many application domains. Data mining is a process of extracting or discovering useful knowledge in large-scale data [4]. It also refers to knowledge discovery from data (KDD) [5], which describes the typical process of extracting useful information from raw data. According to a survey [6] by the program committee members of SIGKDD (the ACM International Conference on Knowledge Discovery and Data Mining), ICDM (the IEEE International Conference on Data Mining), and SDM (the SIAM International Conference on Data Mining), some top data mining topics include link mining, classification, clustering, association analysis and statistical learning. The top algorithms such as k-means, SVM, EM algorithm, kNN, PageRank etc. have been developed to cover all these data mining topics in data mining research [6]. Other data mining techniques include association rule mining, anomaly detection, feature selection and dimension reduction. Additional details can be found in [4, 5, 7, 8, 9]. Data mining is an integral part of many related fields including statistics, machine learning, pattern recognition, database systems, visualization, data warehousing, and information retrieval [5].

Previous study[1] has shown that approximately 80%-85% of all data stored in databases are texts. With the rapid growth of World Wide Web, there are billions of pages containing rich information of images, videos, but most importantly, text contents on the Internet. How to leverage data mining techniques to automatically discover useful information from these unstructured text data has become more and more interesting and challenging. Text mining refers to the process of extracting or mining knowledge from large amount of text data. It usually involves the process of four steps: structuring the input text, defining appropriate distance function between data units, deriving interesting patterns within the structured data, and finally evaluation and interpretation of the mining output such as clusters and classifiers. For examples, commercial search engines such as Google and Yahoo! highly leverage text mining techniques to retrieve relevant documents from user queries, which shows the success of text mining for effective information extraction from massive amount of text data.

During the last decade, the emergence of online social networking sites such as Facebook and Twitter has led to a massive volume of user-generated contents on the Web. Millions of users connect to each other, express themselves and share interests through social networks [10]. Social media, defined as a group of Internet-based applications that build on the technological foundations of Web 2.0, has become a popular platform for news dissemination, professional networking, social

[1] `http://www.edbt2006.de/edbt-share`

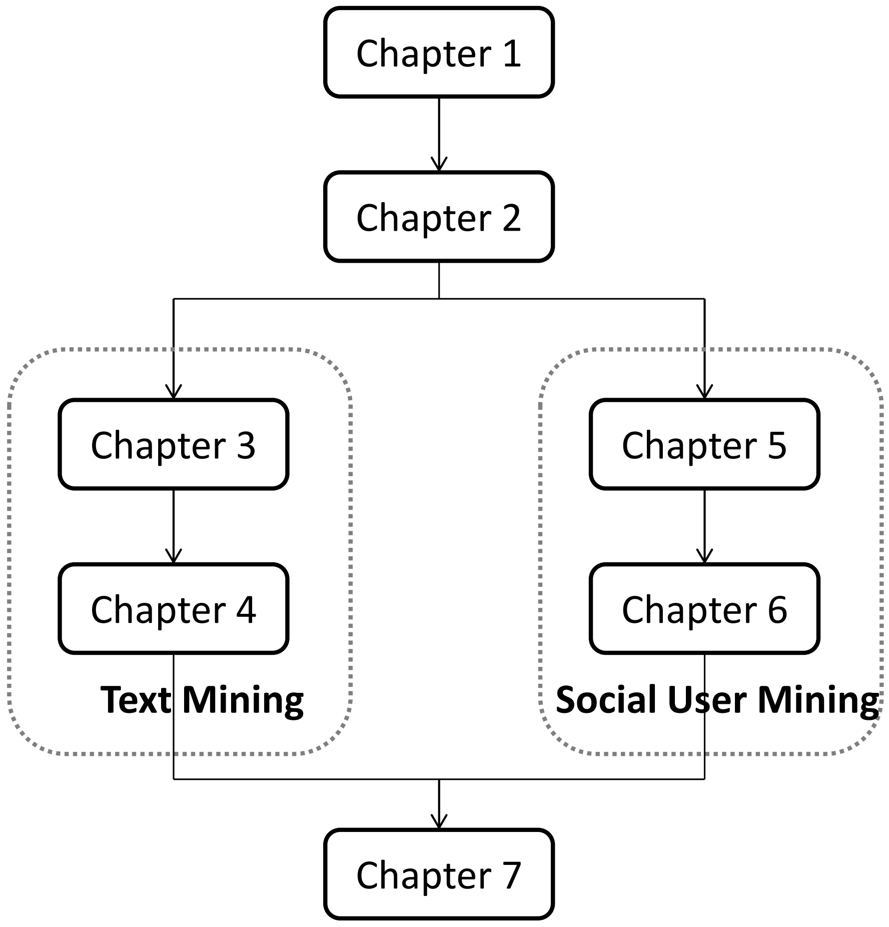


Figure 1.1. Dissertation Overview.

recommendations, and online content curation. The popularity of social media continues to grow exponentially, including blogs, microblogs, social bookmarking applications, location-based social networks and business review sites, etc. Social data, unlike traditional text data, are being created and driven by social users behind the contents. Therefore, social user mining is gaining increasing attention as it appears to become an important vehicle for delivering better user experience.

In this dissertation, we present a series of new methods for knowledge discovery in both text mining and social user mining. Figure 1.1 illustrates an overview of the thesis structure.

The first part of this study is mining *texts* in record linkage (Chapter 3) and document modeling (Chapter 4).

In chapter 3, we study if the mapping between text and time series data is feasible such that the traditional record linkage problem in text mining can find their counterparts in time series (and vice versa). We present a new time series framework that utilizes different combinations of text granularity (e.g., character or word level) and n-grams (e.g., unigram or bigram). To assign appropriate numeric values to each character, our method adopts different space-filling curves (e.g., linear, Hilbert, Z orders) based on the keyboard layout. We achieve comparable accuracy with considerable speed-up in applying our approach to data linkage.

In chapter 4, we introduce two probabilistic generative document models based on latent Dirichlet allocation (LDA), to deal with textual errors in the document modeling problem. Our method is inspired by the fact that most large-scale text data are machine-generated and thus inevitably contain many types of noises. Our new models are developed from the traditional LDA by adding binary switch variables into the term generation process in order to tackle the issue of noisy text data. We show our approach can achieve better generalization performance on real and synthetic OCR document collections.

The second part of this study is mining *users* on social content (Chapter 5) and social activity (Chapter 6).

In chapter 5, we study the problem of mining social content for classifying Twitter user interests. We extract time series features from tweets by exploiting important temporal information and solve the classification problem in time series domain. Our approach is inspired by the fact that some Twitter users often exhibit periodicity patterns when they share their interests or express their opinions. We apply our proposed methods to classification of both sports and political interests of

Twitter users and compare the performance against the traditional classification methods using textual features. We show our time series approach can boost classification accuracy significantly.

In chapter 6, we study the problem of mining social activity for predicting demographical attributes of Facebook users. We introduce a user-level LDA model approach to extract topic features from Facebook Likes. By semantically modeling the relationship between users and their social activity, we show our approach can improve prediction accuracy of both binary and multi-class attributes effectively.

1.2 Motivation

1.2.1 Record Linkage

Since modern database applications increasingly need to deal with dirty data due to a variety of reasons, e.g., data entry errors, heterogeneous formats, or ambiguous terms, recently considerable research efforts have focused on the record linkage problem in order to determine if two entities in a text collection are approximately the same or not [11]. Thus, record linkage with the objective of improving the data quality of database systems has become an important and practical problem.

On the other hand, more and more time series data are being generated from many scientific or application domains, e.g., bio-medical or geographic experiments, monitoring or detection of network traffics, daily fluctuations of stock prices, to name a few. As a consequence, time series data mining has recently received tremendous attention in the data mining community. A lot of new methodologies have been developed for indexing, classification and clustering of time series data [12]. These emerging methods take sequences of numeric values as the subject domain and mainly focus on dimensional reduction, representation methods and distance measures when dealing with time series.

Traditional data mining research focus on document indexing, classification, clustering, association rule mining and record linkage in the context of pure alphabet text which, in turn, needs string manipulation as a supporting technique. Most contemporary data mining techniques take sequences of strings as the subject domain, however in some cases, the text data are information-sensitive and may not be available due to security and privacy reasons. For example, hospitals may want to do data mining or record linkage by third party, but do not want to disclose raw text data of patient records. To tackle this information-sensitive issue, we shift our vision from text to time series when we develop mining methods in the particular situation. Our idea is motivated by the fact that there is lack of connection between the emerging time series data mining approaches and the traditional text mining approaches, and new techniques developed in one area do not easily get carried over to the other area. However, there exist common characteristics in many data mining problems between these two domains. Taking classification as an example, both document classifiers and time series classifiers involve defining appropriate similarity functions in order to find common patterns thereafter. Motivated by these observations, we believe that once we are able to find an effective and efficient way to convert between strings and time series, many relevant data mining problems in one domain can be solved in the other domain. Towards this effort, as a solution to information-sensitive text mining, we aim to build a generic framework to convert string data to time series data.

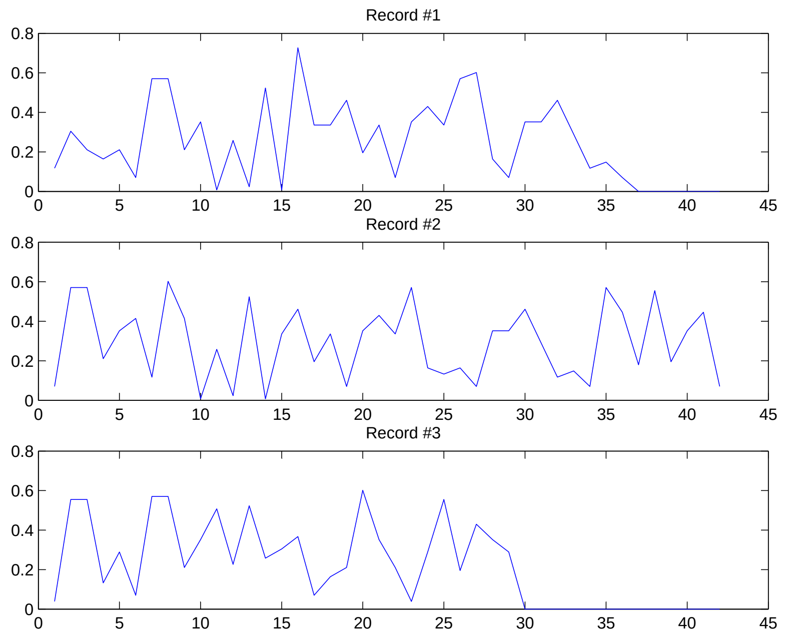
Example 1. We illustrate our idea using a simple example. Suppose we have the following three partient records of name and address information:

Record #1: Steve Allen 15201-B Burbank Bl. Van Nuys CA

Record #2: Allen, S., 15201 Burbank Blvd Van Nuys California

Record #3: Woody Allen 930 5th Ave. New York NY

(a) Original string records



(b)Time series after transformation

Figure 1.2. A simple example of transforming text to time series.

“Steve Allen 15201-B Burbank Bl. Van Nuys CA”

“Allen, S., 15201 Burbank Blvd Van Nuys California”

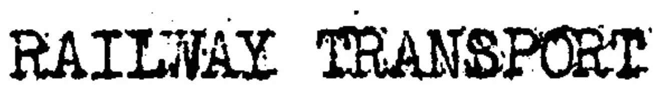
“Woody Allen 930 5th Ave. New York NY”

Obviously, the first two records are referring to the same patient and the third record belongs to a different patient. Figure 1.2 shows the three time series after we use one of the transform schemes of our framework to convert these strings. We can easily see that the time series of the first two records preserve similar shapes in real-value domain (with some shifting) while the time series of the third record has a rather different shape. □

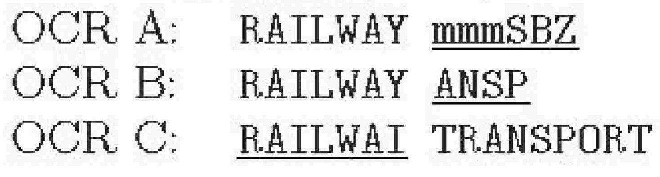
As the first work of this dissertation, we develop a generic framework to convert string data to time series data, such that all the relevant data mining problems in one domain can find their counterparts in the other. We are interested in whether and to what extent the performance of mining solutions developed in the time series domain can be improved over the solutions in the original text mining domain. Toward this objective, we introduce our T^3 (***T***ext ***T***o ***T***ime series) framework to map string/text data to time series data. During the transformation of the entire text corpus, T^3 utilizes different combinations of granularity (i.e., character level or word level) to extract text units from record strings. Furthermore, T^3 utilizes n-grams (i.e., unigram, bigram or trigram) to form subsequences of text units. In order to assign appropriate numeric values to each character, T^3 adopts different space-filling curves (i.e., linear, Hilbert, Z orders) based on the keyboard layout. In addition, to associate real values to each token/word, T^3 uses the tf-idf weight of the traditional weighting scheme from information retrieval and text mining. We apply the T^3 framework to the record linkage problem, one of the traditional data mining problems, to determine whether or not two entities represented as relational records are approximately the same. Through extensive experiments using both real and synthetic data sets, the efficacy of our proposed schemes is experimentally validated.

1.2.2 Document Modeling

Using topic models for representing documents has recently been an area of tremendous interests in text mining and machine learning. Probabilistic topic models are stochastic models for text documents that explicitly model topics in document corpora. Because probabilistic topic models are "generative", they describe a pro-


RAILWAY TRANSPORT

(a) typewritten text


OCR A: RAILWAY mmmSBZ
OCR B: RAILWAY ANSP
OCR C: RAILWAI TRANSPORT

(b) OCR output

Figure 1.3. Three examples of erroneous OCR outputs for a poor quality typewritten text (taken from [1]). Erroneous outputs are underlined.

cedure for generating documents using a series of probabilistic steps. One of the popular paradigms of topic models, characterized by the **Latent Dirichlet Allocation (LDA)** model, consists of a series of probabilistic document models and extensions where topics are modeled as hidden random variables. The LDA model is a widely used Bayesian topic model which can model the semantic relations between topics and words for document corpora. The LDA model assumes that text documents are mixtures of hidden topics and applies Dirichlet prior distribution over the latent topic distribution of a document having multiple topics. In addition, it assumes that topics are probability distribution of words and words are sampled independently from a mixture of multinomials. Since the LDA model was introduced in [13], it has quickly become one of the most popular probabilistic document modeling techniques in data mining and also has inspired a series of extensions (e.g., [14, 15, 16, 17, 18, 19, 20]).

Despite tremendous advancement in document modeling, however, we believe that two major limitations still remain in generative models.

First, *the LDA model assumes that the entire document corpus is error-free to ensure accurate calculation of frequencies of words.* However, an increasing number of new large-scale text data are often machine-generated, and thus inevitably erroneous. For instance, speech recognition softwares can turn audio data into textual transcripts with varying error rates. Similarly, Optical Character Recognition (OCR) engines, despite great success in recent attempts such as Google Books or Internet Archive, are not without problems, and often produce error-abundant text data. [21] pointed out that although researchers are having increasing levels of success in digitizing hand-written manuscripts, error rates remain significantly high. Consider our illustrations below.

Example 2. As an illustration, consider Figure 1.3 that shows three examples of OCR outputs for a poor-quality typewritten text "RAILWAY TRANSPORT." All three popular OCR engines (i.e., ABBYY FineReader, OmniPage Pro, and Google Tesseract) generated outputs with one erroneous word for each. It is known that the accuracy of the LDA model often declines significantly as word error rates increase [1]. Now, consider Table 1.1 that shows some top words (selected by the LDA model) for five topics of a small sample of `Unlv`[2] OCR data set. From the list, we can see that there exist a lot of erroneous words in the selected top words. In addition, the words are not representative and the differences between the topics are difficult to identify. This example shows that the performance of traditional LDA model greatly suffers when documents contain erroneous words. □

Second, *since the LDA model does not consider the order of the topics and words, during parameter estimation and inference, the topics and the words are assumed to be exchangeable.* The LDA model relies on the bag-of-words document prototype. It assumes each word in a document is generated by a latent topic and explicitly models the word distribution of each topic as well as the prior

[2]`http://code.google.com/p/isri-ocr-evaluation-tools/updates/list`

Table 1.1. Top words (selected by LDA) for five different topics of a small sample of `Unlv` OCR data set (erroneous words are in italic).

Top words
school, *stu*, district, teacher, angel, lo, board, *educ*
sto, food, *res*, *servic*, low, leonard, *temperatur*, retail
air, *airlin*, *fli*, american, *engin*, subject, *threate*, *pil*
mln, *dlrs*, year, net, quarter, share, *dlr*, *ln*
mcknight, vista, *de*, fleetwood, brown, *davi*, san, *democr*

distribution over topics in the document. However, we argue that the ordering of words and phrases are often critical to capture the meaning of texts in data mining tasks. Successive words in the same document are more likely to belong to the same topic. For example, a phrase “social network” is a term in modern information society under Web 2.0 while “social” is a term from traditional sociology and “network” refers to a particular term in computer science. Often, the ordering of terms carries special meanings in addition to the appearance of individual words. Therefore, incorporating topic dependency is important to learn topics and also to disambiguate words which may belong to different topics. More importantly, considering the ordering of consecutive terms can often help in dealing with errors found in parts. For instance, despite the typo “betwork” in the middle from a phrase “social betwork analysis”, surrounding correct words “social” and “analysis” still have common semantic connections that could be exploited.

As the second work of this dissertation, we introduce our novel models to tackle the issues of noisy text data in document modeling. In particular, we propose a new LDA model termed as TE-LDA (LDA with ***T***extual ***E***rrors) to deal with textual errors in document corpora. We further extend it to a new TDE-LDA (LDA with ***T***opic ***D***ependency and textual ***E***rrors) model in order to take into account topic dependency in the document generation process. Through a set of comprehensive experiments, the efficacy of our proposed models is validated using both real and synthetic data sets.

1.2.3 Social User Mining

The emergence and rapid growth of online social media sites such as Facebook and Twitter has led to a massive volume of user-generated contents on the Web. For example, since it is founded in 2006, Twitter has grown to become one of the most popular social media and microblogging services. Twitter users can share timely social information about daily or weekly activities and statuses with their followers by posting short text messages (i.e., tweets) with 140 characters limit in length. As of 2012, there are over 500 millions of registered users generating over 340 millions of tweets every day[3].

With a growing number of users using social network services (SNS), being able to understand about the users better becomes critical in many applications. If businesses have an access to users' demographics such as gender and ethnicity or particular user interests toward brand or sports teams, such knowledge can be useful in personalizing the contents or targeted online marketing [22]. Despite these implications, however, only a small fraction of users voluntarily provides information about themselves. For example, Twitter as one of the most popular microblog sites has been used as a rich source of real-time information sharing in everyday life. Many popular search engines such as Google, Yahoo! and Bing have started including feeds from Twitter in their search results. When Twitter users express their interests about organizations, companies, brands, or sports in tweets, it in turn provides important opportunities for businesses in improving their services such as targeted advertising and personalized services. Since the majority of Twitter users' basic demographic information (e.g., gender, age, ethnicity) is unknown

[3]`http://techcrunch.com/2012/07/30/page/2/`

or incomplete, being able to accurately identify the hidden information about users becomes an important and practical problem. However, Twitter provides limited metadata about their users. Important user profile information such as age and gender are typically incomplete or inaccurate. Moreover, other useful attributes such as ethnicity, personal properties and preferences are not usually disclosed due to privacy concerns [23]. Consequently, observable user information such as contents of tweets provides valuable and more importantly reliable information for social user mining and knowledge discovery in Twitter.

Some researchers have already studied the problem of Twitter user classification [23, 24, 25]. For example, [23] presented an exploratory study of classification about latent user attributes in Twitter including gender, age, regional origin, and political orientation. The authors utilized various features including sociolinguistic features like presence of emoticons, unigrams and bigrams features derived from the tweet text, and other behavior and network features for binary classification. [25] focused on other binary classification problems such as political affiliation (e.g., Democrats or Republicans), ethnicity identification and affinity for business like Starbucks. The authors developed a machine learning framework to learn classification models from labeled data and a broad set of features such as profile features, tweeting behavior features, linguistic content features and social network features.

However, previous research has shown that profile information do not carry enough good-quality information to be directly applied for user classification purposes [24]. Moreover, other information such as tweeting behavior features are not useful for most classification tasks [23]. Different from previous work, we focus on classifying Twitter users based on the *contents* of their tweets. While linguistic contents usually assume the bag-of-words model on textual data, we argue that Twitter users often exhibit a *periodicity* pattern when they post tweets to share

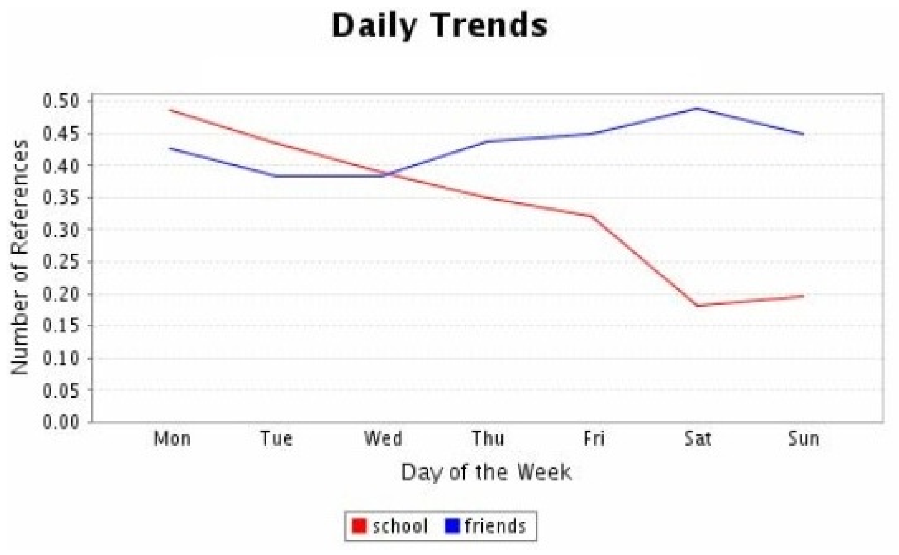


Figure 1.4. Daily trends for the terms “friends” and “school,” taken from [2].

their activities and statuses or express their opinions. This is because people tend to show interests in different activities during different time frames. Recent research [26] has shown that contents on microblogging platforms such as Twitter show patterns of temporal variation and pieces of content become popular and fade away in different temporal scales. Previous study [2] has shown that contents on microblogging platforms such as Twitter show patterns of temporal variation and there exists a recurring pattern in word usage. Such patterns may be observed over a day or a week. For example, Figure 1.4 (taken from [2]) shows that the terms “school” and “friends” have different frequency patterns during weekly time frame.

Example 3. Consider Figure 1.4 that shows the trends for the terms “friends” and “school.” Note that the term “school” is more frequent during the early week and “friends” takes over during the weekend. □

As a result, instead of using tweet messages directly, one may leverage the *temporal* information derived from the word usage within tweet streams and model tweet features as *time series* to amplify its *periodicity* pattern to boost the accuracy in classification. Therefore, in classifying Twitter users, we advocate to convert tweet contents into time series based on word usage patterns, and then perform time series based classification algorithms.

As the third work of this dissertation, we introduce our novel time series approach to tackle the problem of classifying Twitter user interests. In particular, we propose a new framework to convert Twitter users to time series by incorporating temporal information into the stream of tweets such that the user classification problem can be solved in time series domain. The efficacy of our proposed approach is validated through extensive experiments in both *sports* and *political* interest domains.

On the other hand, among the vast amount of user-generated contents, Facebook Likes activity is one of the highly available and public information in online social networks. Facebook Likes refer to the social activity by Facebook users to express their positive association with online contents such as photos, friends' status updates, products, sports, musicians, books, restaurants, or other popular websites [27]. Compared to other social activities [28], Facebook Likes are currently publicly available by default. Previous research has shown that 57% of Facebook user profiles publicly reveal at least one Like among different categories. This large amount of available activity information suggests that the majority of users consider this Facebook activity does not violate their privacy as there seems no correlation between their Likes and private data.

Thus, it is natural to try to utilize Facebook Likes provided by users in order to infer the missing user attributes in an online social network. Such ability of automatically predict user attributes is very useful for many social networking applications such as friend recommendation and content sharing. Also it has less privacy concerns as users are more willing to publicly reveal their Likes activity in online social networks.

As the fourth work of this dissertation, we introduce our topic modeling approach to tackle the problem of attribute prediction of Facebook users. In particular, we propose a LDA approach to extract the topic features from Facebook Likes activity such that the prediction problem can be solved using latent topics. The efficacy of our proposed approach is validated through extensive experiments using real data sets on four *binary* and *multi-class* demographical attributes of Facebook users.

1.3 Dissertation Organization

The rest of this dissertation is organized as follows.

In Chapter 2, we review the related work for record linkage, document modeling and social user mining.

In Chapter 3, we present our novel idea of T^3 framework on mining texts using time series in order to solve the *record linkage* problem in time series domain. We propose two variations of granularity, three variations of n-grams, and four variations of score assignments based on space-filling curve techniques for characters or tf-idf weighting technique for tokens, in order to convert record strings. We show the efficacy of our proposed T^3 schemes using both real and synthetic data sets.

In Chapter 4, we present our TE-LDA and TDE-LDA model on mining texts using latent topics in order to tackle the *document modeling* problem in the presence of textual errors. We incorporate textual errors into term generation process in TE-LDA. We further extend it to TDE-LDA by taking into account topic dependency to leverage on semantic connections among consecutive words even if parts are typos. Through extensive experiments, we show that our proposed models are able to model the document corpus in a more meaningful and realistic way, and achieve better generalization performance than the baseline LDA model and the n-grams model.

In Chapter 5, we study the problem of *mining social content* for classifying Twitter user interests using tweet messages. We apply the traditional document categorization methods to Twitter user classification in both binary and multi-class scenarios. We further introduce a novel time series approach to convert Twitter users to time series by incorporating temporal information into the stream of tweets such that the user classification problem can be solved in time series domain. Moreover, we propose two classification algorithms for multi-class user classification in time series domain. Through extensive experiments, we demonstrate that our time series approach can boost classification accuracy significantly with respect to identifying Twitter users with certain sports and political interests.

In Chapter 6, we study the problem of *mining social activity* for predicting Facebook user attributes using Facebook Likes. We introduce a user-level LDA model approach to tackle the problem of user attribute prediction. In particular, we extract topic features from Facebook Likes activity such that the prediction problem can be solved using latent topics. Through comprehensive experiments, we show that our proposed approach can improve prediction accuracy effectively using real data sets on four binary and multi-class demographical attributes of Facebook users.

Finally, we conclude this dissertation by summarizing our new methods and contributions, and outline future work in Chapter 7.

Chapter 2

Related Work

2.1 Record Linkage

The general linkage problem has been known as record linkage [29, 30], merge-purge [31], citation matching [32], object matching [33], entity resolution [34], authority control [35], and approximate string join [33, 36], to name a few. Excellent survey papers [37, 11] provide the latest advancement of the linkage problem. In terms of scalability issue, researchers have proposed the blocking technique, computationally efficient distance function [38], or parallel linkage [39]. In addition, [38] conducted an in-depth study on the blocking issue of the linkage problem in the context of digital library. [40] presented the novel idea of solving the record linkage problem using BLAST, one of the most popular gene sequence alignment algorithms in Bioinformatics. They proposed four variations of linkage solutions to translate text data into DNA sequences and demonstrated the good combination of accuracy and speed of applying BLAST to record linkage.

On the other hand, time series data mining has received tremendous attention in the data mining community during the last decade. Many time series representation methods such as Discrete Fourier Transformation (DFT) [12], Discrete Wavelet Transformation (DWT) [41], Piecewise Aggregate Approximation (PAA) [42], Singular Value Decomposition (SVD) [12] and Symbolic Aggregate approXimation (SAX) [43] etc. have been proposed together with the corresponding similarity measures such as Euclidean Distance (ED) [12], Dynamic Time Warping (DTW) [44], Distance based on Longest Common subsequence (LCSS) [45] and so on. Recently, [46] summarized and evaluated the state-of-the-art representation methods and similarity measures for time series data through extensive experiments.

However, none of these existing works attempted to solve the linkage problem using time series mining techniques as we did in this dissertation. Recently the authors in [47] mentioned a method to transform text into a time series representation in the case of translating biblical text in both English and Spanish. The basic idea is to convert the bible text into bit streams based on the occurrences of a particular word in the text. Then a time series is generated based on the number of word occurrences within a predefined sliding window across the bit streams. Although it is useful in the case of generating time series for the translation versions of the same text in two different languages, their method can not been directly applied to the record linkage problem because each record may have different sets of words and it would be hard, if not impossible, to find a common word among them before the time series conversion.

2.2 Document Modeling

Probabilistic document modeling has received tremendous attention in the data mining community. A series of probabilistic models have been introduced to simulate the document generation process. These models include the Naive Bayesian model and the Probabilistic Latent Semantic Indexing (PLSI) model [48]. The LDA model has become most popular in the data mining and information retrieval community due to its solid theoretical statistical foundation and promising performance. A wide variety of extensions of LDA model have been proposed for different modeling purposes in different contexts. For example, the author-topic model [14, 18] uses the authorship information with the words to learn topics. The correlated LDA model learns topics simultaneously from images and caption words [15]. The Link-LDA model and Topic-link LDA model [16] represent topics and author communities using both content words and links between documents.

Most topic modeling techniques require the bag-of-words assumption [13]. They treat the generation of all words independently from each other given the parameters. It is true that these models with the bag-of-words assumption simplified the problem domain and enjoyed a big success, hence attracted a lot interests from researchers with different backgrounds. Some researchers tried to drop this assumption to assume that words are generated dependently. For example, [49] developed a bigram topic model on the basis of the hierarchical Dirichlet language model, by incorporating the concept of topic into bigram models. [50] proposed a topical n-grams model to automatically determines whether to form an n-gram based on the surrounding context of words. [19] developed a probabilistic time series model to capture the evolution of topics in large document corpora. [51] proposed a hidden topic Markov model (HTMM) to incorporate a hidden Markov structure into LDA. However, their model is based on the assumption that all words in the same sentence must have the same topic and imposes a sentence boundary for words. [52] proposed a correlated topic model which allows for correlations between topic assignments and draws a topic proportion from a logistic normal instead of a Dirichlet distribution. [53] proposed the HMMLDA model as a generative composite model which considers both short-range syntactic dependencies and long-range semantic dependencies between words. [54] proposed a probabilistic model to match documents at both general topic level and specific word level in information retrieval tasks.

Recently, a number of researchers proposed topic segmentation models which are closely related to topic models. Topic segmentation is to split a text stream into coherent and meaningful segments. For example, the aspect hidden markov (HMM) model proposed in [55] models unstructured data which contains streams of words. In the aspect HMM model, documents are separated into segments and each segment is generated from a unique topic assignment and there is no mixture of topics during the inference. [56] proposed a Bayesian approach to linear topic segmentation which assumes some numbers of hidden topics are shared across multiple documents. [57, 58] further extended this work by marginalizing the language models using the Dirichlet compound multinomial distribution, and applied the model to both linear topic segmentation and hierarchical topic segmentation for the purpose of multi-scale lexical cohesion. [59] proposed a statistical model that combines topic identification and segmentation in text document collections, and the model is able to identify segments of text which are topically coherent and cluster the documents into overlapping clusters as well. Note that the Markov transition is based on segments with each being generated from a linear combination of the distributions associated with each topic.

Most topic modeling techniques require clean document corpora. This is to prevent the models from confusing patterns which emerge in the noisy text data. Recent work in [1] is the first comprehensive study of document clustering and LDA on synthetic and real-word Optical Character Recognition (OCR) data. The character-level textual errors introduced by OCR engines serve as baseline document corpora to understand the accuracy of document modeling in erroneous environment. As pointed out by these researchers, even on clean data, LDA will often do poorly if the very simple feature selection step of removing stop-words is not performed first. The study shows that the performance of topic modeling algorithms degrades significantly as word error rates increase.

2.3 Social User Mining

Recently, researchers tried to tackle the problem of short text classification in social networks from different perspectives [60, 61, 62]. [60] used a small set of domain-specific features extracted from the user's profile and text to effectively classify the text to a predefined set of generic classes such as news, events, opinions, deals and private messages. [61] tried to improve the classification accuracy by gaining external knowledge discovered from search snippets on Web search results. [62] proposed a non-parametric approach for short text classification in an information retrieval framework. And the predicted category label is determined by the majority vote of the search results for better classification accuracy. [63] proposed a classification model of tweet streams that switches between two probability estimates of words, which can learn from stationary words and also respond to busty words. Note that these classification methods are carried over *tweets*. In contrast, in this dissertation, we focus on the problem of classification over *users*.

Several researchers have investigated the problem of detecting user attributes such as gender, age, regional origin, political orientation, sentiment, location, and spammer based on user communication streams. [64] investigated statistical models for identifying the gender of Twitter users as the binary classification problem.

They adopted a number of text-based features through various basic classifier types. [23] presented a study of classification experiments about more latent user attributes such as gender, age, regional origin, and political orientation. They adopted various sociolinguistic features such as emoticons, ellipses, character repetition, etc., and used support vector machines to learn a binary classifier. Although the authors gave a general framework with classification techniques for various user attributes mining tasks, they employed a lot of domain knowledge in their experiments. [65] focused on classification problem on positive or negative feelings on tweet streams for opinion mining and sentiment analysis. [66, 67] studied user geo-location detection problem in the city level. Based purely on the contents of the user's tweets, the authors proposed a probabilistic framework to automatically identify words in tweets with a local geo-scope for estimating a Twitter user's city-level location. [68] further improved the prediction quality of a Twitter user's home location by estimating the spatial word usage probability with Gaussian Mixture models. Meanwhile, they also proposed unsupervised measurements to rank the local words to remove noises effectively. [69] used a number of characteristics features related to user social behavior as attributes of machine learning process for classifying users as either spammers or non-spammers on Twitter.

[70] proposed a temporal semantic analysis model to compute the degree of semantic relatedness of words by studying patterns of word usage over time. [26] proposed a time-aware clustering algorithm to uncover the common temporal patterns of online content popularity. [24] is closely related to our work. The authors developed a general machine learning approach to learn three binary classification models based on Decision Trees for identifying political affiliation, ethnicity, and business affinity from labeled data using a broad set of features such as profile, tweeting behavior, linguistic content and social network features.

On the other hand, some researchers have studied the problem of predicting private traits and attributes from digital records of human behavior such as browsing logs [71, 72, 73], properties of Facebook or Twitter profiles including the number of friends or the density of friendship networks [74, 75, 76, 77]. Recent study [27] has shown that Facebook Likes can be used to automatically and accurately predict sensitive personal attributes, such as sexual orientation, ethnicity, religious and political views, intelligence, happiness, drug use, parental separation, age, and gender. Based on demographic profiles and results of psychometric tests as well as their Facebook Likes data from 58,466 volunteers, their study used regression models to predict individual psychodemographic profiles from Facebook activities. The method can achieve best discriminative results for predicting dichotomous variables such as gender and ethnicity. And the authors claimed that even for numerical variables such as openness attribute from personality traits, the prediction accuracy is also close to the accuracy of a standard personality test.

Chapter 3

Record Linkage Using Time Series

3.1 Overview

In this chapter, we focus on the *Record Linkage* problem, one of the traditional data mining problems, to determine whether or not two entities represented as relational records are approximately the same. The record linkage problem frequently occurs in database applications (e.g., digital libraries, search engines, customer relationship management) and get exacerbated in the situation of integrating data from heterogeneous sources as poor quality data is prevalent in databases due to a variety of reasons including lack of standards for recording database fields, transcription errors, data entry mistakes, etc. For instance, a customer address table in a data warehouse may contain multiple address records which are all from the same residence, and therefore need to be consolidated. For another example, suppose we would like to integrating two digital libraries such as DBLP and CiteSeer. Because citations in two systems tend to have different formats, it is not always trivial to identify all matching pairs.

Formally, the record linkage problem can be formulated as follows.

Definition (Record Linkage) *Given a collection of record entities R, identify all similar entity pairs (a, b) such that $a, b \in R$ and $dist(a, b) < \phi$, where $dist()$ is some distance function and ϕ is some pre-selected threshold.* $\square$

Despite significant advancement in each area, data mining research in textual data (e.g., web pages of search engines, citations of digital libraries, relationship data in social networks) and time series data (e.g., network traffic observations, daily fluctuations of stock prices) have not been developed in a close synchronization. New techniques developed in one area do *not* easily get carried over to the other area. This is partly due to the fact that although both deal with many similar problems such as defining appropriate distance functions or finding interesting patterns, their subject domains are different – i.e., alphabetical strings vs. numerical signals. Therefore, toward this lack of connection between the emerging time series and the traditional text mining approaches, in this chapter, we investigate if there exists feasible transformation between two data types such that relevant data mining problems in one data type can find their counterparts in the other type. We are interested in whether and to what extent the performance of mining solutions developed in one domain can be improved over the solutions in the other domain.

In this chapter [78], we present the **T^3** (***T***ext ***T***o ***T***ime series) framework to map text data to time series data. During the transformation of the entire text corpus, T^3 utilizes different combinations of granularity (i.e., character level or word level) to extract text units from strings. Furthermore, T^3 utilizes n-grams (i.e., unigram, bigram or trigram) to form subsequences of text units. In order to assign appropriate numeric values to each character, T^3 adopts different space-filling curves (i.e., linear, Hilbert, Z orders) based on the keyboard layout. In

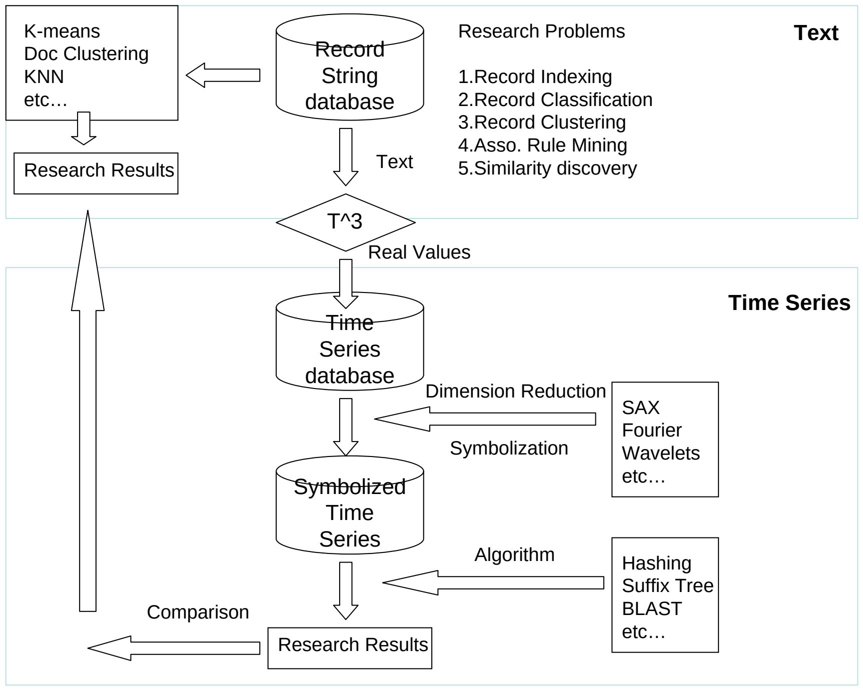


Figure 3.1. Overview of the T^3 framework.

addition, to associate real values to each token, T^3 uses the tf-idf weight of the traditional weighting scheme from information retrieval and text mining. We apply the T^3 framework to the record linkage problem. Through extensive experiments using both real and synthetic data sets, the efficacy of our proposed schemes is experimentally validated. To the best of our knowledge, this is one of the first attempts to solve a text mining problem in time series domain. Our experiments reveal that T^3 shows comparable accuracy (despite the *lossy* transformation in T^3) when compared to a popular distance measure (e.g., Levenshtein distance) in text domain. However, T^3 also achieves much improved speed-up thanks to the numerical data of time series domain.

3.2 Proposed T^3 Framework

The basic idea of T^3 is illustrated in Figure 3.1. Instead of solving data mining problems on string/text data, we first scan the string database using the proposed transformation schemes in T^3. After the string database is mapped to a new time series database, we then employ dimension reduction and symbolization techniques directly on the real values of time series. In general, T^3 serves as a convenient bridge to connect two subject domains: numerical signals and alphabetical strings. Therefore, our approach can be considered as a novel complement to existing text mining algorithms which were solely built for generic use based on string manipulation.

In this section, we introduce in detail the transformation schemes in our proposed T^3 framework. Given any sequence s from a database of textual sequences D, T^3 utilizes different combinations of granularity, n-grams and score assignments to convert strings to time series. In particular, T^3 performs three basic steps during the transformation:

- **Phase 1 (Determine the text unit)**: T^3 can treat each record or document in the database as a sequence of characters or a sequence of word tokens. At character-level granularity, each alphabet, number or special character in the sequence s is considered as a single text unit. And at word-level granularity, a single unit would be any word in the sequence s.

- **Phase 2 (Determine the n-grams)**: After determining the single unit of the sequence, T^3 adopts the n-grams concept from various areas of statistical natural language processing. Basically, an n-gram is a sub-sequence of n single units from a given sequence. In particular, in T^3, an n-gram is a sub-sequence of n consecutive characters at character-level granularity and a sub-sequence of n consecutive tokens at word- level granularity. T^3 uses three kinds of n-grams to transform the entire text corpus.

- **Phase 3 (Determine the score assignment)**: Given the units and n-grams, in the third step, T^3 now turns to assign appropriate numeric values to convert the sequence s to a time series. Based on different granularity, T^3 uses different weighting techniques to assign scores to each unit.

We explain the detailed schemes of transformation in the following.

3.2.1 Granularity

Each record or document in the text domain can be considered as a sequence of characters or a sequence of word tokens. During the process of converting string data into time series data, T^3 extracts single units from sequences and then assign pre-assigned values to each unit. In this chapter, we use two different levels of granularity as follows:

- *character level*: An alphabet letter is regarded as a single text unit. In the transformation, ignoring upper/lower cases, we consider 64 (= 26 + 10 + 28) cases – i.e., 26 cases for 26 English alphabets, 10 cases for 10 numbers (e.g., 0 to 9), and 28 cases for all special characters such as `@`, `#`, `$`. We do distinguish among special characters since some of special characters in record strings appear in our data sets.

- *word level*: At this granularity, an English word (also called "token") is regarded as a single text unit and T^3 simply extracts each token from sequences and then assign appropriate values to each token based on the weighting scheme. Note that token-level granularity is coarse compared to fine granularity at character level. Therefore, for the same text sequence, the length of time series after transformation at token level is much shorter than the length of corresponding time series after transformation at character level.

In the experimentation, by default, we use the character-level granularity.

3.2.2 N-grams

In statistical natural language processing, an n-gram is a sub-sequence of n consecutive items from a given sequence. These items could be symbols, letters, or words according to the application. As mentioned above, in our T^3 framework, we treat either a character or a token as the single unit of sequences. Therefore, an n-gram in T^3 is a sub-sequence of n consecutive characters at character level and a sub-sequence of n consecutive tokens at word level. n-grams have different sizes and T^3 adopts three sizes of n-grams to extract sub-sequences from the original text sequence as follows. We illustrate these three n-grams using a sequence “time series data mining” as an example.

- *unigram*: An n-gram of size 1 is called a unigram. That is, it only contains one single unit of the text sequence. For the above example, a unigram at word level would be “time”, “series”, “data”, “mining”.
- *bigram*: An n-gram of size 2 is called a bigram. That is, it contains two single units of the text sequence. For the above example, a bigram at word level would be “time series”, “series data”, “data mining”.
- *trigram*: An n-gram of size 3 is called a trigram. That is, it contains three single units of the text sequence. For the above example, a trigram at word

Table 3.1. A complete example of different combinations of granularity level and n-grams.

Coding	Transformation
`char + unigram`	"t","i","m","e","s","e","r","i","e","s","d", "a","t","a","m","i","n","i","n","g"
`char + bigram`	"ti","im","me","es","se","er","ri","ie","es","sd", "da","at","ta","am","mi","in","ni","in","ng"
`char + trigram`	"tim","ime","mes","ese","ser","eri","rie","ies","esd", "sda","dat","ata","tam","ami","min","ini","nin","ing"
`word + unigram`	"time", "series", "data", "mining"
`word + bigram`	"time series", "series data", "data mining"
`word + trigram`	"time series data", "series data mining"

level would be "time series data", "series data mining".

In the experimentation, by default, we use unigram during the transformation.

Example 4. Table 3.1 shows a complete example of how to transform the sequence "time series data mining" based on different combinations of granularity level and n-grams. □

3.2.3 Score Assignment

Score assignment is the most important part of the T^3 framework. At this stage, T^3 assigns appropriate numeric values in order to actually convert strings to time series. Based on different levels of granularity, T^3 adopts different weighting schemes in order to assign scores to subsequences of text units.

At character level, T^3 uses the QWERTY keyboard layout, to allocate each text unit, i.e. alphabets, numbers or special alphabets, into equal-length or varying-length bins within the range [0,1]. Then the median value of each bin is used to represent the corresponding character. During the allocation of bins, we consider the following three possible layouts:

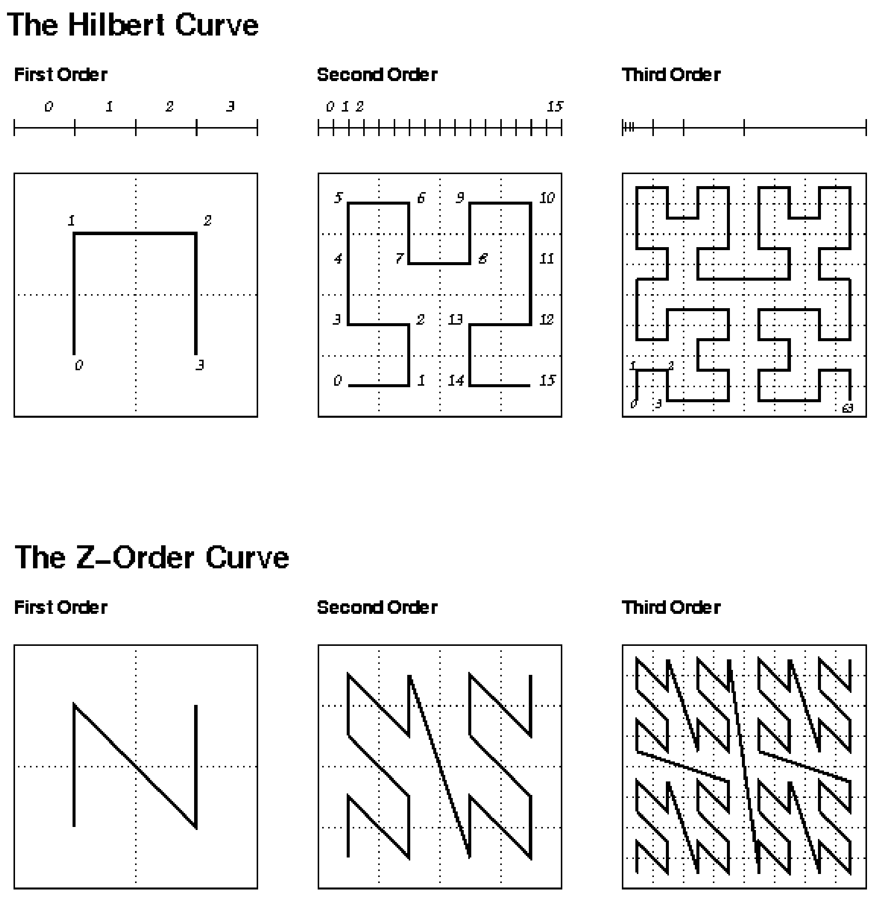


Figure 3.2. Hilbert curve vs. Z-order curve.

- *linear order*: This layout is simply based on each key position on the keyboard following the order of row by row. Each character is then assigned an appropriate real value within the range [0,1].

- *Hilbert order*: In this layout, we regard the keyboard as a small 2D space and then adopt the space filling curve techniques to map 2-dimensional space to 1-dimensional sequence. Figure 3.2(top) shows the details of Hilbert space-filling curve. After we get the 1-dimensional sequence, we then allocate it to uniform bins within the range [0,1] so that each character is assigned a real value. Hilbert order has good locality-preserving behaviors so that alphabets at similar locations in the keyboard have similar real values during the score assignment. Our idea is motivated by the fact that alphabets at similar locations in the keyboard have higher probability of typo, which is a traditional problem in data mining research. Using Hilbert space-filling curve, characters at similar locations will have similar real values which, in turn, mitigate the typo problem that may cause sequence dissimilarity.

- *Z order*: In addition to Hilbert space-filling curve, we also implement Z order space-filling curve. Figure 3.2(bottom) shows the details of Z space-filling curve. Z order also has good locality-preserving behaviors compared to Hilbert order and we are interested in whether there is significant performance difference between these two space-filling curves.

Table 3.2 shows the actual assignment of scores using linear, Hilbert, and Z-ordering.

At word level, T^3 uses the tf-idf weight (term frequency-inverse document frequency) of the traditional weighting scheme from information retrieval and text mining. The tf-idf weight is a statistical measure to estimate how important a token in a string record is within a record database such that it increases proportionally to the number of times that the token occurs in the string but is offset by its frequency in the database. Each token of a record string is assigned an importance weight using the tf-idf weight such that the whole string can be converted into a time series.

3.2.4 Discussion

Note that the three dimensions of approaches (i.e., granularity, n-gram, and score assignment) in T^3 are not exhaustive at all. One can easily devise more sophis-

Table 3.2. Actual assignment table.

Char	Linear	Hilbert	Z
A	0.5535	0.1215	0.2295
B	0.9045	0.5805	0.7155
C	0.8505	0.2565	0.3915
D	0.6075	0.2295	0.3375
E	0.3375	0.3645	0.1755
F	0.6345	0.3105	0.3645
G	0.6615	0.5535	0.6615
H	0.6885	0.6345	0.6885
I	0.4725	0.7695	0.6345
J	0.7155	0.6615	0.7695
K	0.7425	0.7425	0.7965
L	0.7695	0.9855	0.9855
M	0.9585	0.6885	0.8235
N	0.9315	0.6075	0.7425
O	0.4995	0.9585	0.9315
P	0.5265	0.9315	0.9585
Q	0.2835	0.0945	0.0675
R	0.3645	0.3375	0.2025
S	0.5805	0.2025	0.2565
T	0.3915	0.5265	0.4995
U	0.4455	0.7965	0.6075
V	0.8775	0.2835	0.4185
W	0.3105	0.0675	0.0945
X	0.8235	0.1755	0.3105
Y	0.4185	0.4995	0.5265
Z	0.7965	0.1485	0.2835
0	0.2565	0.9045	0.9045
1	0.0135	0.0135	0.0135
2	0.0405	0.0405	0.0405
3	0.0675	0.3915	0.1215
4	0.0945	0.4185	0.1485
5	0.1215	0.4455	0.4455
6	0.1485	0.4725	0.4725
7	0.1755	0.8235	0.5535
8	0.2025	0.8505	0.5805
9	0.2295	0.8775	0.8775
All special char.	0.9855	0.7155	0.8505

ticated transformation schemes from text strings to numeric time series. For instance, as to the score assignment dimension, in addition to the keyboard layout based assignment for the character level or weighting based assignment for the word level, one may use Linguistic characteristic (e.g., while a character-level bigram "on" occurs frequently, another bigram "xz" rarely occurs in English) to assign different assignment scores. Similarly, domain-specific characteristics of text data can be adopted. For instance, instead of character-level or word-level, one may use phrase-level or paragraph-level summary as the basic text unit when dealing with documents. Since our immediate goal is to evaluate the validity of T^3 framework to show that "*there exist some reasonable information-lossy conversion schemes from text domain to time series domain so that text-based data mining problems can be solved in time series domain*", we rather leave the development of more sophisticated conversion schemes in T^3 framework for future work.

3.3 Experimental Validation

In order to validate our proposed T^3 framework, we use the *record linkage* problem. In a nutshell, once we transform all textual records into time series data using T^3 framework, for a given query time series q, we attempt to retrieve q's true duplicate time series. Then, we compare the performance of T^3 with that of a traditional record linkage solution that uses the text string as input. If the performance of T^3 in solving the record linkage problem in time series domain is comparable to that of a traditional record linkage solution, then it shows the validity of our proposed T^3 framework. Since the transformation schemes in T^3 "lose" some information of original text string (i.e., lossy conversion), we expect the accuracy of T^3 framework to drop slightly, compared to the accuracy of a traditional record

Table 3.3. Summary of data sets.

Name	Data	Error	Domain	# of records	Max # of duplicates	# of queries	# of targets
`map`	real	real	map name	337	2	19	19
`bird`	real	real	bird name	982	2	67	67
`business`	real	real	business name	2,139	2	279	279
`census`	real	real	census info.	841	2	326	326
`university`	real	real	university name	116	16	15	15
`cora`	real	real	citation	1,326	5	98	194
`restaurant`	real	real	restaurant info.	864	2	111	111
`celebrity`	real	real	celebrity address	2615	2	276	276
`dblp`	real	synthetic	citation	5359	5	1,369	3,991
`dbgen`	synthetic	synthetic	mailing list	9,947	19	960	8,987

linkage solution. However, what we are more interested in these experimentations is the comparison among different schemes in T^3 framework and any possible benefits of those schemes.

The record linkage problem can be modeled as a *selection* problem (i.e., select top-k similar records) as well as a *threshold* problem (i.e., find all similar records above a threshold). For illustration purpose, we briefly describe the selection version of the linkage problem as follows. Imagine that given a query record q_r, there are k true duplicates, d_1, ..., d_k, in the database of records D_r. Then, the goal of record linkage is to obtain the true duplicates from the retrieved top-k records. In the best scenario, if all k true duplicates are retrieved, then we get the precision and recall of 1.0. Note that in this chapter, we adopt both the selection approach and the threshold approach for further experimental analysis.

Given a set of query records Q_r and a database of records D_r, a naive nested-loop style algorithm with quadratic running time to find all duplicate records is:

```
for each record $r \in Q_r$
    for each record $s_i \in D_r$
        compute $dist(r, s_i)$;
    return $\{s_1, ..., s_k\}$ with the lowest $dist(r, s_i)$.
```

3.3.1 Set-Up

Table 3.3 shows the summary of data sets that we used in our experiments. The first five data sets `map`, `bird`, `business`, `census`, and `university` are real data sets which contain *real* string data and *real* errors. They are downloaded from Second String[1] data repository. Each of the original data sets has several fields with the first field as the key attribute of each record. We pre-processed the data sets by finding the true duplicates for each query record and then manually removed all the key fields to make each record only contain information of its own domain. The other three data sets `cora`, `restaurant`, and `celebrity` are also real data sets containing real string data and real errors. Both `cora` and `restaurant` data sets are downloaded from RIDDLE[2] data repository[3]. Note that the original `cora` data set from RIDDLE has 1,295 citation records of computer science papers, which are all duplicates of 116 unique citations. Each citation has a number of duplicates, some as many as 55 duplicates. Each citation record is labeled with a unique identifier. We hand-selected 292 records of 98 citations such that each citation can have up to five duplicates. We did not use the full data set in the original `cora` data set since some of the duplicates were incorrectly labeled and it is unrealistic that duplicates of only a few entities take a large portion of the entire database records. Since the number of selected records is too small, we next extended the data set with 1,034 additional citation records manually collected from the Web (mainly from the paper lists of researchers in computer science departments), making total of 1,326 records. From here forward, we refer to this modified version of `cora` data set as simply `cora` data set. In the experiments, we search for 194 duplicates of

[1]http://secondstring.sourceforge.net/
[2]http://www.cs.utexas.edu/users/ml/riddle/
[3]The celebrity data set was provided to us by Ned Porter at US Census Bureau.

Table 3.4. Parameter settings used for the `dbgen` (above) and `dblp` (bottom) data set. Note that probabilities for different error types are independent to each other.

Error Type	Probability
Typo errors exist	0.3
Single typo error	0.6
Multiple typo error	0.4
First name is dropped	0.3
First name is abbreviated	0.7
Middle name is dropped	0.2
Middle name is abbreviated	0.8
Typo errors exist	0.9
Single typo error	0.9
Multiple typo error	0.1
Venue information as a full name	0.4
Pages and month information are dropped	0.5
Numeric information has a typo	0.2

98 original records in this `cora` data set.

Also note the original `celebrity` data set has 2764 records of celebrities' addresses with their names. The name of a celebrity is used as a key to distinguish duplicates in the data set. In the original data set, there are the records with the same address for different celebrity names. They may refer the addresses shared by family members who are all celebrities. Since such records are not proper for our problem, only one of them are kept among the records with the same address for different names so that 129 records are removed from the original `celebrity` data set. And, among the remaining 2615 records, 276 records are used as query strings and each of them has an exactly one duplicate.

Next, a new citation data set `dblp` was generated using real citation records from similar venues of DBLP. To inject artificial errors, we randomly selected ten venues from similar research domains such as CIKM, SIGKDD, SIGMOD, and again randomly selected citations published in those venues. Once we had initial citations, we generated duplicates by introducing typographical errors. For

Table 3.5. Examples of some data sets.

Name	Sample Data
`bird`	"Gavia stellata Red-throated Loon"
`business`	"3Com Corporation"
`restaurant`	"cassells 3266 w sixth st la 213 480 8668 hamburgers"
`celebrity`	"ANDRE AGASSI 8921 ANDRE DR. LAS VEGAS NV 89113"
`dblp`	"Bell Data Modelling of Scientific Simulation Proams SIGMOD Conference 1982"
`dbgen`	"Colbri P Beer 478 Naftel St 6j2 Rio Blanco PR 00744"

instance, single typo error was generated by inserting a new character, replacing a characters by different character, deleting a character, or swapping two characters in a word. However, for numeric data fields (e.g, journal volumes or numbers), we only use insertion and deletion. The detailed parameter settings to generate errors are shown in Table 3.4.

Finally, using the data generation tool, DBGen [31], we generated one synthetic data set containing patient information with nine attributes such as SSN, first name, middle name, last name, street number, street name, city, state, and ZIP number. Note that unlike aforementioned data sets, this data set contains only synthetically generated data and errors. DBGen permits us to control error rates in records in detail. Using DBGen, we generated the synthetic patient data – `dbgen`. `dbgen` has around 10,000 patient records with on average 19 duplicates per record. In `dbgen`, the probability for a token to have a typo (10% single vs. 90% multiple typos) in person and street name fields is substantially increased to 0.8.

In order to get a general idea, Table 3.5 shows examples of record strings in some of the data sets.

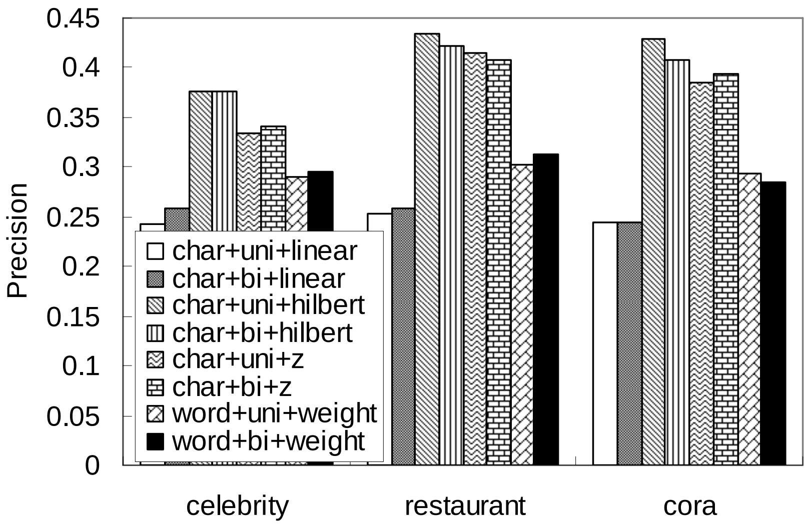
Distance Measures. Given two time series T_1 and T_2, a distance function, denoted by $Dist(T_1,T_2)$, calculates the distance between the two time series. we use the *Levenshtein Distance* (LD) in text domain and *Euclidean Distance* (ED) and *Dynamic Time Warping* (DTW) in time series domain. All measures are known to work well for order-conscious text or time series data [44]. Since ED requires two time series to have the same length, in our experimentation, we augment the shorter time series to have the same length as the longer time series by simply adding prefix or suffix of median values (i.e., 0.5). The DTW function, on the other hand, allows a time series to be "stretched" or "compressed" to provide a better match with other time series [47, 44]. In this chapter, we show the performance of these two distance functions in terms of precision, recall and running time.

3.3.2 Evaluation Metrics

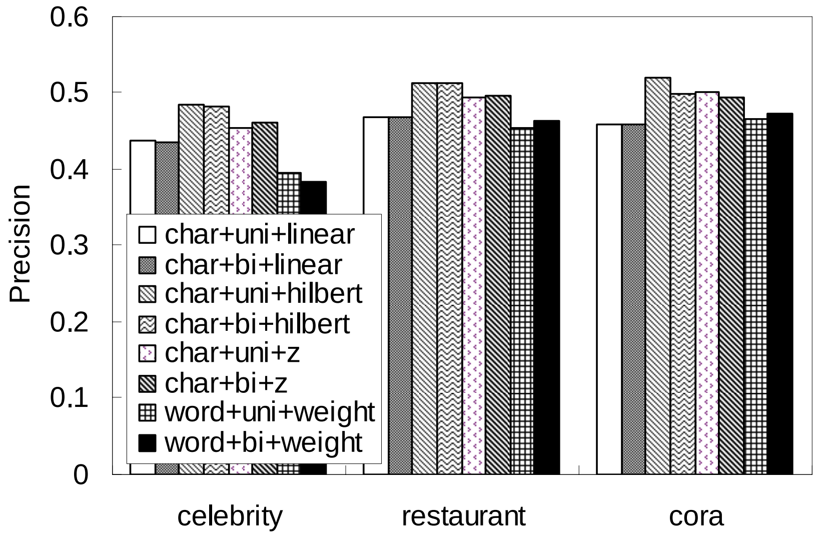
To evaluate the efficiency and effectiveness of the proposed T^3 framework, we mainly use two evaluation metrics – speed and accuracy. In particular, to measure the speed of a method, we use the *Running Time* (T) excluding any pre-processing steps. To measure the accuracy, we use the average *Precision* (P) and *Recall* (R). Suppose that T denotes a set of true matching records and S denotes a set of records retrieved by an algorithm. Then, we have: precision=$\frac{|S \cap T|}{|S|}$ and recall=$\frac{|S \cap T|}{|T|}$. We will use the precision-recall (PR) graph to present the accuracy.

3.3.3 Comparison of Transformation Schemes

We first compare the performance of different combinations of granularity, n-grams and score assignment in T^3. In this comparison, we choose one distance function (i.e, either ED or DTW) and then perform tests of all major coding schemes using the same distance measure. Figures 3.3(a) and (b) present the precision of the record linkage task using ED and DTW, respectively. Among data sets, the results of tests on `celebrity`, `restaurant`, and `cora` are presented. In Figure 3.3(a) with ED, note that both Hilbert and Z order based schemes outperform the others with



(a) Comparison using ED



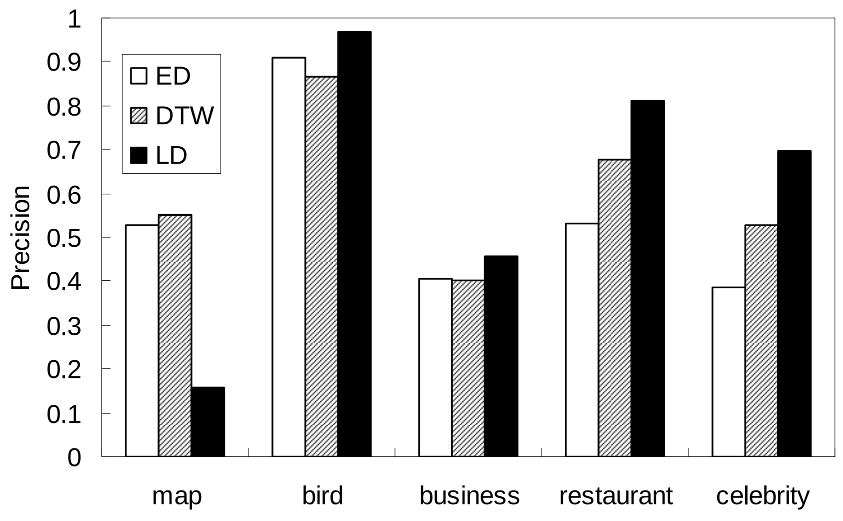
(b) Comparison using DTW

Figure 3.3. Comparison among eight different transformation schemes based on distance function ED and DTW using three data sets.

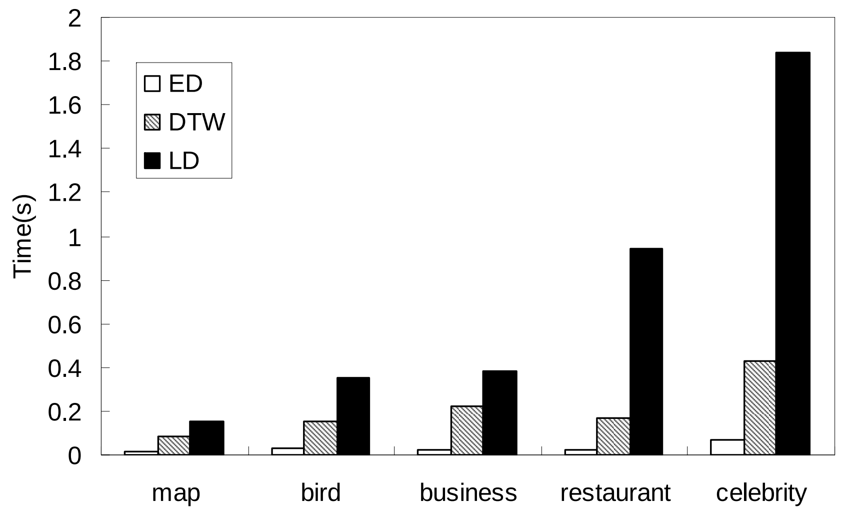
respect to the precision. This is reasonable because both Hilbert and Z order have a good locality-preserving behavior such that alphabets in the neighborhood in the keyboard have the similar real values during the score assignment. Therefore, this can reduce a number of false positives in cases when true duplicates have some dissimilar characters caused by typos or data entry errors. Between these two orders, Hilbert order performs slightly better than Z order scheme, but not significantly. Figure 3.3(b) shows the similar results when DTW is adopted as the distance function in the experiment.

Another interesting finding is that the word-level transformation schemes using tf-idf weighting as scores do *not* show a significantly better precision, although they can find true duplicates faster because of the shorter time series generated. The reason is that using the word-level schemes based on tf-idf weights, the resulting time series is entirely determined by the tf-idf weight of each token. To some extent, we lose the lexical information of tokens. For instance, there might exist a situation where two tokens are completely different but happen to carry equal or similar tf-idf weights. This can affect the shapes of time series and hence generate false positives.

Also note that from Figures 3.3(a) and (b), we do not see much difference between unigram and bigram schemes (trigram schemes have similar patterns and not shown for limited space). This is partially because our record linkage solutions are obtained in real-value domain after record stings are converted to time series, and higher-gram techniques may not be as effective as in the case of string manipulation in text domain. Overall, the transformation scheme based on the combination of character-level granularity, unigram and Hilbert order appears to be the best scheme. Therefore, we adopt this scheme (denoted as `char-uni-hilbert`) in the following experiments.



(a) Precision



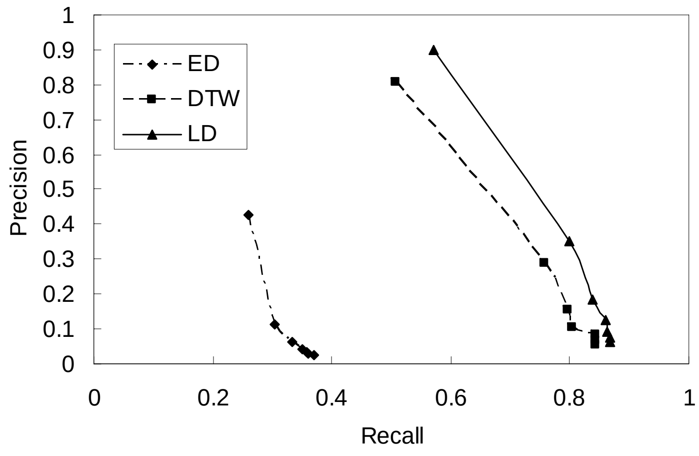
(b) Running time per query (in sec.)

Figure 3.4. Comparison among ED, DTW, and LD using five data sets (based on `char-uni-hilbert`).

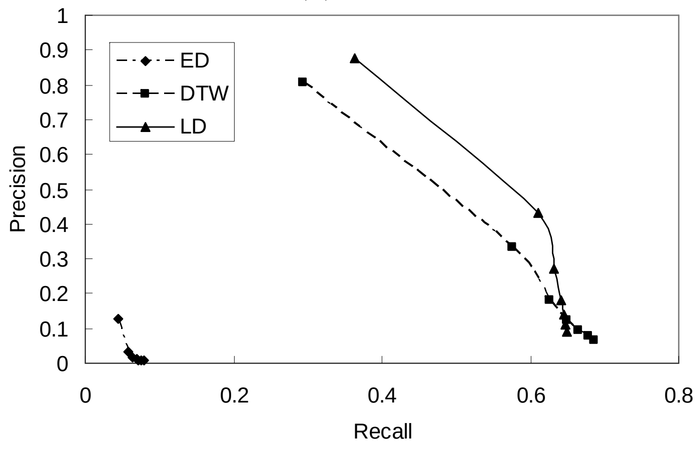
3.3.4 Comparison of Distance Functions with Baseline

Next, we compare the performance of different distance functions in our proposed T^3 framework. In this comparison, we fix the transformation scheme to `char-uni-hilbert` and then compare among ED and DTW (in time series domain), and LD (in text domain) as the baseline. Figures 3.4(a) and (b) show the precision and running time of three distance measures in the context of record linkage problem. The results of tests on `map`, `bird`, `business`, `restaurant` and `celebrity` are presented. In these data sets, each query string has exactly one duplicate. Therefore record linkage on these data sets is straightforward and aims to find the other duplicate for each of the query records.

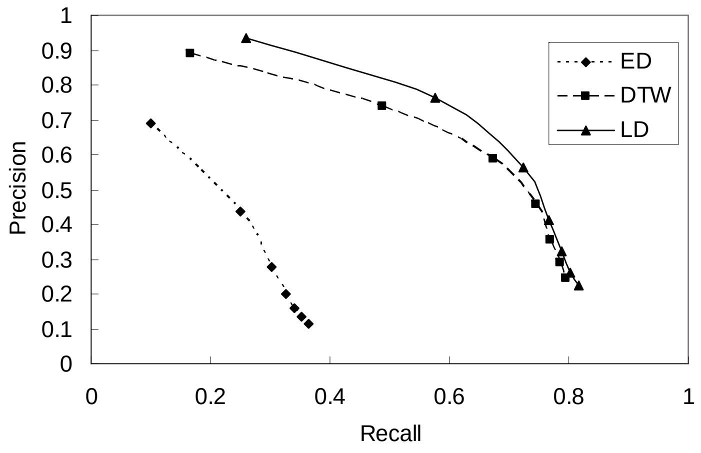
In Figure 3.4(a), note that LD consistently produces better precision than ED and DTW, except `map` data set. This is as expected because LD directly operates on the original record strings in text domain without the loss of any information. What we are more interested is: *as a complementary approach for solving the record linkage problem in time series domain, how good is the performance of T^3 techniques compared to the baseline?* Figure 3.4(a) shows that T^3 with ED and DTW can yield comparable precision on four data sets (and better precision on one data set). Since the various transformation schemes in T^3 tend to lose some information from original text strings during the conversion, we expect to lose some degree of accuracy in T^3, when compared to LD. Therefore, although there appears to be degraded accuracy in T^3, since it is comparable to the baseline without using T^3, we believe that the result is still promising. Also note that the overall precision of either our proposed schemes or the baseline is around 0.6 across all the data sets in Figure 3.4(a). This is due to the characteristics of our data sets. As shown in Table 3.5, our data sets are real data sets which contain a lot of mis-spelling errors and mis-alignments. Therefore, we expect low degree of accuracy of matching similar records. More research is needed so that one can transform text to time series while maintaining or improving the accuracy. Another interesting finding is that DTW mostly performs better than ED (i.e., in `map`, `restaurant`, and `celebrity` data sets) and the difference increases as the size of data sets and



(a) cora



(b) dblp



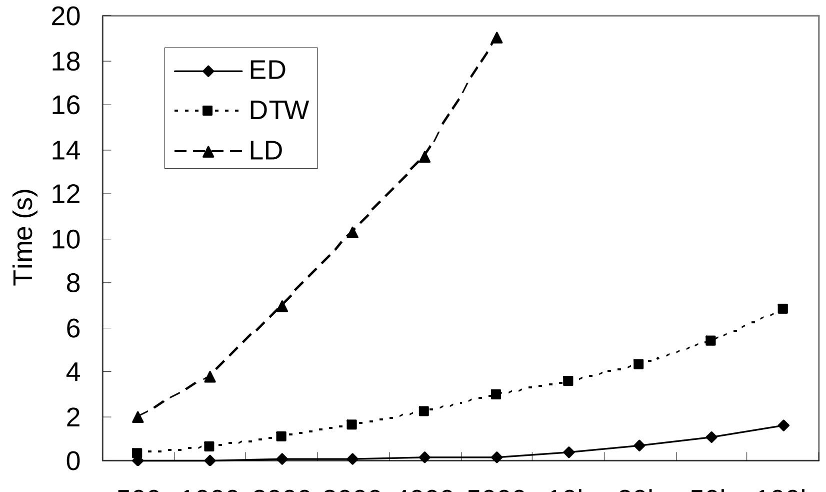
(c) dbgen

Figure 3.5. PR graphs of ED, DTW and LD methods using `char-uni-hilbert`.

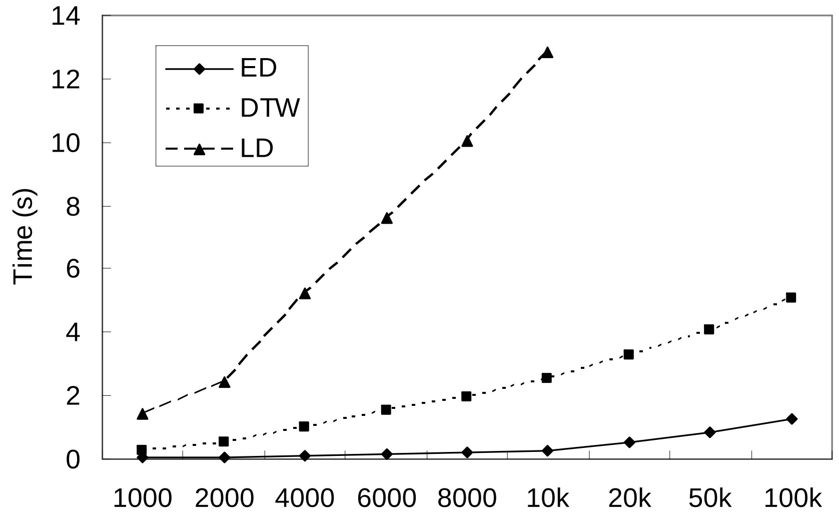
lengths of record strings increase. This is reasonable as DTW is usually regarded as a much more robust distance measure for time series [44] and allows similar shapes to match even in the case that two time series are not aligned well in the time axis.

Figure 3.4(b) shows the average running time per query. We can clearly see that both ED and DTW methods consistently run much faster than the baseline LD, across all data sets. Furthermore, ED method even performs faster than DTW. While ED is the fastest (with low accuracy), overall, DTW shows a good trade-off of having faster running time with comparable accuracy. The running time of DTW increases as the length of record strings increases. This is partially due to the cost of dynamic programming in DTW, which is in the magnitude of square of record length.

The precision-recall (PR) graphs in Figure 3.5 are generated with both precision and recall of the ED, DTW and LD methods by increasing k, which is the number of answers returned by an algorithm to solve the record linkage task. The value of k changes from 1 to 30. At each point, corresponding precision and recall values are measured and plotted. The PR graphs of three large data sets, `cora`, `dblp` and `dbgen`, are presented. As we can see from Figure 3.5, DTW and the baseline LD outperform ED by a large margin. Furthermore, DTW produces PR curves that are comparable to the baseline. This is consistent with what we found in Figure 3.3(c). In addition, note that both LD and DTW run much faster than LD (in Figure 3.6), again consistent with Figure 3.3(d).



(a) Scalability on `dblp`



(b) Scalability on `dbgen`

Figure 3.6. Running time per query (s) on different sizes of `dblp` and `dbgen` subsets.

3.3.5 Scalability

Since the scalable processing is a critical issue in the record linkage problem, we also study the scalability of T^3 framework. We select two large data sets, i.e., `dblp` and `dbgen`. From `dblp` with 5,359 real citation records, we prepare different subsets of 500, 1000, 2000, 3000, 4000 and 5000 records. Furthermore, we generate four different datasets of 10,000, 20,000, 50,000 and 100,000 records from the original `dblp` dataset. Also, from `dbgen` with 9,947 mailing lists, we generate subsets of 1000, 2000, 4000, 6000, 8000, and 10000 records (we add 53 more mailing lists to generate 10000 records). Again, we generate three different datasets of 20,000, 50,000 and 100,000 records from the original `dbgen` dataset. With these data sets and the fixed transformation scheme of `char-uni-hilbert`, we measure the running time as the data size increases.

Figure 3.6 shows the results. In general, Figures 3.6(a) and (b) show similar patterns. As the size of data sets increases, the running time per query increases linearly for both ED and DTW. However, the running time for LD increases more rapidly compared to that of our approaches. This indicates that our record linkage solution is more scalable to handle a large amount of data. Furthermore, ED consistently outperforms DTW in terms of speed. This is as expected because DTW involves a procedure of dynamic programming in calculating the distance, which decreases the overall speed as the size of data increases. But as we mentioned earlier, DTW method has better precision in terms of accuracy of record linkage.

3.4 Summary

In this chapter, we have presented our novel design of the T^3 framework to transform text to time series data. We proposed two variations of granularity, three variations of n-grams, and four variations of score assignments based on space-filling curve techniques for characters or tf-idf weighting technique for tokens. We adopted two similarity measures, Euclidean Distance (ED) and Dynamic Time Warping (DTW), to calculate the distance between two time series and have shown the efficacy of our proposed schemes using both real and synthetic data sets.

In terms of record linkage, our schemes in the T^3 framework show promising results with good combination of speed and accuracy, compared to conventional string matching methods such as Levenshtein Distance (LD). In particular, Hilbert space-filling technique at character-level granularity is the best variation of transformation schemes while DTW is a better distance measure regarding precision and ED outperforms regarding running time. With respect to accuracy and speed, the experimental results confirm that our T^3 framework can generate precision-recall curves comparable to the baseline LD. We believe our approach can shed new insights in both areas of text mining and time series mining.

Chapter 4

Document Modeling Using Latent Topics

4.1 Overview

In this chapter, we focus on the *Document Modeling* problem, one of the traditional document representation and text retrieval problems. The most conventional methodology proposed for documents is to represent each document as a weight vector of numbers in vector space models [79], each of which is a tf-idf weight of each word in the document. The tf-idf weight is based on the frequency of a word in a document and tend to give a measure of the importance of the word to represent the document in a corpus. Language models [80] are also used to represent documents, such as bigram, instead of unigram as in tf-idf method. Recently some more sophisticated ways for document modeling are emerging, such as topic models. Probabilistic topic models are stochastic models for text documents, which explicitly model topics in the document corpus. As generative models, they describe a procedure for generating documents using a series of probabilistic steps.

Since it was introduced in 2003 [13], the latent Dirichlet allocation (LDA) model has quickly become a powerful tool for statistical analysis of text documents. LDA assumes that text documents are mixtures of hidden topics and applies Dirichlet prior distribution over the latent topic distribution of a document having multiple topics. Also, it assumes that topics are probability distribution of words and words are sampled independently from a mixture of multinomials. Therefore, LDA is a widely used Bayesian topic model which can model the semantic relations between topics and words for document corpora.

Formally, the document modeling problem can be formulated as follows.

Definition (Document Modeling) *Given a document collection D from a fixed vocabulary V, model and extract a set of T topics $\{\phi_1, ..., \phi_T\}$ where ϕ_i is a semantically coherent topic in D defined as a multinomial distribution of all words in V, i.e., $\{p(w|\phi_i)\}_{w \in V}$ with the constraint $\sum_{w \in V} p(w|\phi_i) = 1$.* □

LDA requires accurate counts of the occurrences of words in order to estimate the parameters of the model, Therefore, it assumes that the entire document corpus is clean in order to ensure correct calculation of frequencies of words. However, as text data become available in massive quantities, textual errors are appearing inevitable in large-scale document corpora. These textual errors include typos, spelling errors, transcription errors caused by text or speech recognition tools, digitization errors of Google Books and Internet Archives, etc. For example, Walker et al. [1] point out that although researchers are having increasing levels of success in digitizing hand-written manuscripts, error rates remain significantly high. Figure 1.3 shows an example of typewritten documents and output by three Optical Character Recognition (OCR) engines. Even on clean data, LDA will often do poorly if the very simple feature selection steps of removing stop-words is not performed first. It is shown that the performance in terms of accuracy declines significantly as word error rates increase [1], which highlights the importance of taking into account the noisy data issue in document modeling.

As large-scale text data become available on the Web, textual errors in a corpus are often inevitable (e.g., digitizing historic documents). Due to the calculation of frequencies of words, however, such textual errors can significantly impact the accuracy of statistical topic models such as LDA. To address such an issue, in this chapter [81], we propose two novel extensions to LDA (i.e., TE-LDA and TDE-LDA): (1) The TE-LDA model incorporates textual errors into term generation process; and (2) The TDE-LDA model extends TE-LDA further by taking into account topic dependency to leverage on semantic connections among consecutive words even if parts are typos.

In summary, with respect to the document modeling problem with varying degrees of *noisy* corpora and using the perplexity as an evaluation metric, our second proposal with a better result, TDE-LDA, outperforms: (1) the traditional LDA model by 16%-39% using real data and by 20%-63% using synthetic data; and (2) the state-of-the-art N-Grams model [50] by 11%-27% using real data and by 16%-54% using synthetic data.

4.2 The LDA Model

In this section, we give a brief overview of the *Latent Dirichlet Allocation* (LDA) model. [13] introduced the LDA model as a semantically consistent topic model, which attracted considerable interest from both the statistical machine learning and natural language processing communities. LDA models documents by assuming that a document is composed by a mixture of hidden topics and that each topic

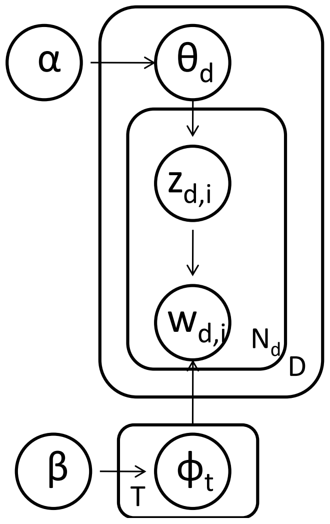


Figure 4.1. The LDA Model.

is characterized by a probability distribution over words.

The model is known as a graphical model for topic discovery. The notations are shown in Table 4.1. θ_d denotes a T-dimensional probability vector and represents the topic distribution of document d. ϕ_t denotes a W-dimensional probability vector where $\phi_{t,w}$ specifies the probability of generating word w given topic t. $Multi(.)$ denotes multinomial distribution. $Dir(.)$ denotes Dirichlet distribution. α is a T-dimensional parameter vector of the Dirichlet prior distribution over θ_d, and β is a W-dimensional parameter vector of the Dirichlet prior distribution over ϕ_t. The process of generating documents is shown in Algorithm 1.

Algorithm 1: **The LDA Model**.

```
1 For each of the $T$ topics $t$, sample a distribution over words $\phi_t$ from a
  Dirichlet distribution with hyperparameter $\beta$;
2 For each of the $D$ documents $d$, sample a vector of topic proportions $\theta_d$ from
  a Dirichlet distribution with hyperparameter $\alpha$;
3 For each word $w_{d,i}$ in document $d$, sample a topic $z_{d,i}$ from a multinomial
  distribution with parameters $\theta_d$;
4 Sample word $w_{d,i}$ from a multinomial distribution with parameters $\phi_{z_{d,i}}$.
```

Performing exact inferences for the LDA model is intractable due to the choice

Table 4.1. Notations

Symbol	Description
D	total number of documents
W	total number of word tokens
T	total number of topics
N_d	total number of words in document d
$w_{d,i}$	ith word in document d
$z_{d,i}$	latent topic at ith word in document d
$\theta_{d,i}$	probability of topic i in document d
$\phi_{t,w}$	probability of word w in topic t

of distribution and the complexity of the model. The existing approximate algorithms for parameter estimation and inference of the LDA model include variational methods [13], EM algorithm [48] and Markov Chain Monte Carlo (MCMC) [82]. One assumption in the generation process above is that the number of topics is given and fixed. LDA model considers documents as "bags of words", i.e., there is no ordering between words and all words as well as their topic assignments in the same document are assumed to be conditionally independent. Furthermore, finding good estimates for the parameters of LDA model requires accurate counts of the occurrences and co-occurrences of words, which in turn requires a "perfect" corpus with errors as few as possible.

4.3 Proposed Models

To account for textual errors in the traditional LDA topic model, in this section, we propose a new LDA model termed as TE-LDA (LDA with ***T***extual ***E***rrors) to take into account noisy data in the document generation process. We further extend it to a new TDE-LDA (LDA with ***T***opic ***D***ependency and textual ***E***rrors) model in order to take into account topic dependency in the document generation process. We explain the details of our proposed models in the following.

4.3.1 TE-LDA

In this model [83], we distinguish the words in the documents and separate them as tokens and typos. Given a document, each word has a probability to be an error and we want to capture this probability structure in the term generation process. In order to reflect the nature of textual errors in the generative model, we adopt a switch variable to control the influence of errors on the term generation.

The proposed model is illustrated in Figure 4.2. Here we introduce some notations used in the graphical model: D is the number of documents, T is the number of latent topics, N_d is the total number of words in document d (with $N_d = N_{term} + N_{typo}$, the sum of all the true terms and typos). α, β and β' are parameters of Dirichlet priors, θ_d is the topic-document distribution, ϕ_t is the term-topic distribution. ϕ_{typo} is the term distribution specifically for typos. We include an additional binomial distribution δ with a *Beta* prior of γ which controls the fraction of errors in documents.

For each word w in a document d, a topic z is sampled first and then the word w is drawn conditioned on the topic. The document d is generated by repeating the process N_d times. To decide if each word is an error or not, a switch variable X is introduced. The value of X (which is 0 or 1) is sampled based on a binomial distribution δ with a *Beta* prior distribution of γ. When the sampled value of X equals 1, the word w is drawn from the topic z_t which is sampled from the topics learned from the words in document d. When the value of X equals 0, the word w is drawn directly from the term distribution for errors. Overall, the generation process for TE-LDA can be described in Algorithm 2.

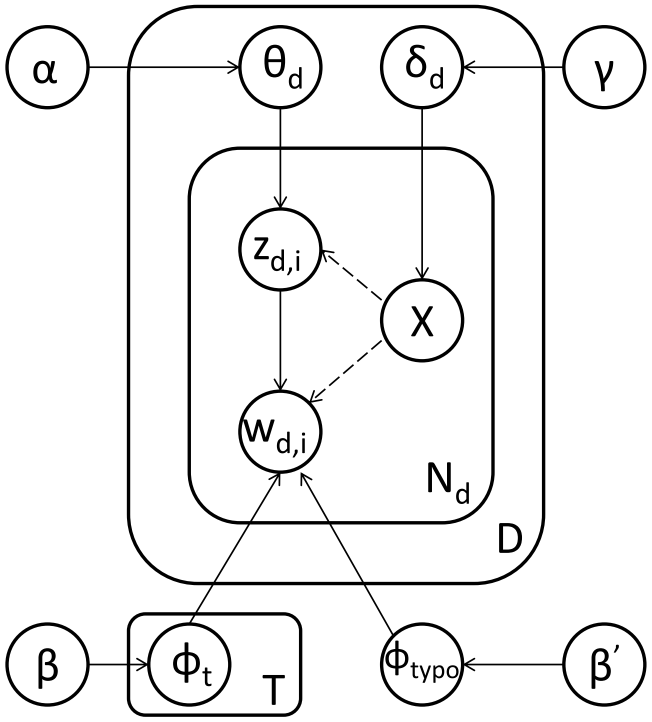


Figure 4.2. TE-LDA Model.

4.3.2 Topic Dependency

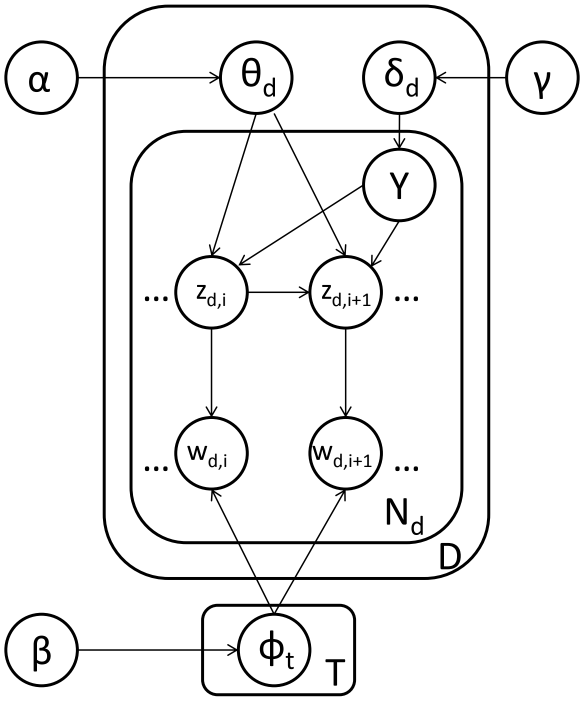
As we mentioned in the introduction section, LDA relies on the bag-of-words assumption. However, in many data mining tasks, words are often connected in nature and successive words in the document are more likely to be assigned the same topic. Therefore, incorporating topic dependency is important to capture the semantic meaning of texts and also to disambiguate words which may belong to different topics. Even in noisy text corpora, consecutive words may be dependent to each other regardless of textual errors. For example, in a phrase "text dat mining" with textual error "dat" as typo of word "data", the correct word "text" and "mining" still have semantic connections and both words belong to the same topic of data mining. Hence, incorporating this correlation gives a more realistic model of the latent topic structure and we expect to obtain better generalization

Algorithm 2: **The TE-LDA Model**.

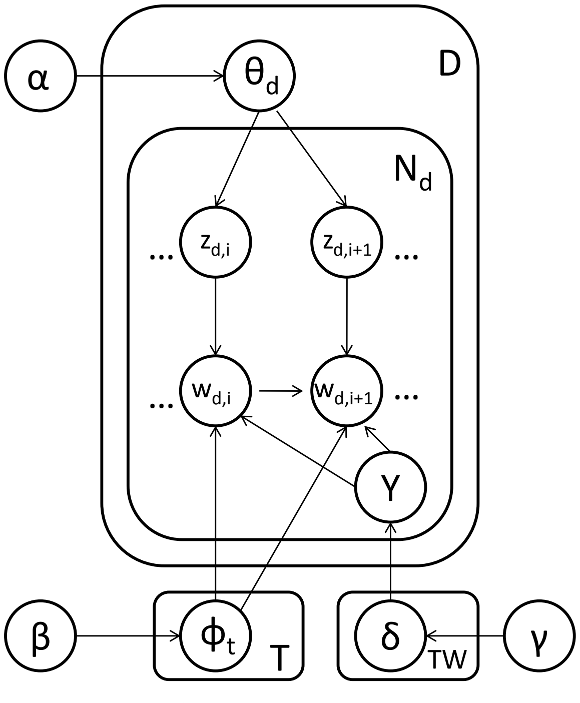
```
1  For each of the $D$ documents $d$ , sample $\theta_d \sim$ Dir($\alpha$)and $\delta_d \sim$ Beta($\gamma$);
2  For each of the $T$ topics $t$, sample $\phi_t \sim$ Dir($\beta$);
3  Sample $\phi_{typo} \sim$ Dir($\beta'$);
4  foreach $N_d$ words $w_{d,i}$ in document $d$ do
5      Sample a flag $X \sim$ Binomial($\delta_d$);
6      if $X$ = 1 then
7          Sample a topic $z_{d,i} \sim$ Multi($\theta_d$);
8          Sample a word $w_{d,i} \sim$ Multi($\phi_{z_{d,i}}$);
9      endif
10     if $X$ = 0 then
11         Sample a word $w_{d,i} \sim$ Multi($\phi_{typo}$);
12     endif
13 endfch
```

performance quantitatively. To apply topic dependency and drop the bag-of-words assumption, we assume the topics in a document form a Markov chain with a transition probability that depends on a transition variable Y. When Y equals 0, a new topic is drawn from θ_d. When Y equals 1, the current topic of word w_i is equivalent to the previous topic of word w_{i-1}.

[50] proposed a topical n-grams model to automatically determine whether to form an n-gram based on the surrounding context of words. The n-grams model is an extension of the bigram topic model, which makes it possible to decide whether to form a bigram for the same two consecutive words depending on the nearby context. As a result, the n-grams model imposes a Markov relation on the word set. In contrast, topic dependency considers the relation between consecutive *topics* instead of words. That is, the Markov relation is on the topic set instead of the word set. Figure 4.3(a) shows an alternative graphical model for applying topic dependency to LDA. The n-grams model is illustrated in Figure 4.3(b). We incorporate topic dependency in our proposed TE-LDA model in the following.



(a) LDA with topic dependency



(b) N-Grams Model

Figure 4.3. Comparison of topic dependency and term dependency.

4.3.3 TDE-LDA

We extend our TE-LDA model and further incorporate topic dependency into one unified model, named as TDE-LDA. The proposed model is illustrated in Figure 4.4.

For each word w in a document d, a topic z is sampled first and then the word w is drawn conditioned on the topic. The document d is generated by repeating the process N_d times. To decide if each word is an error or not, a switch variable X is introduced. The value of X (which is 0 or 1) is sampled based on a binomial distribution δ with a *Beta* prior distribution of γ. When the sampled value of X equals 1, the word w is drawn from the topic z_t which is sampled from the topics learned from the words in document d. To decide if the current topic is dependent to the previous topic or not, a switch variable Y is introduced. The value of Y (which is 0 or 1) is sampled based on a binomial distribution δ with a *Beta* prior distribution of γ. When the sampled value of Y equals 1, the topic z_i is assigned to be identical to the previous one z_{i-1} to reflect the dependency between them. When the value of Y equals 0, the topic z_i is sampled from the topics learned from the words in document d. And the word w is drawn from the topic z_t. When the value of X equals 0, the word w is drawn directly from the term distribution for errors. The generation process for TDE-LDA can be described in Algorithm 3.

4.3.4 Discussion

In this section, we discuss two important issues on our proposed models.

Rare Words vs. Textual Errors

In terms of frequency of words, note that it is difficult to differentiate between rare-but-correct-English words and typos because both appear rather seldom in

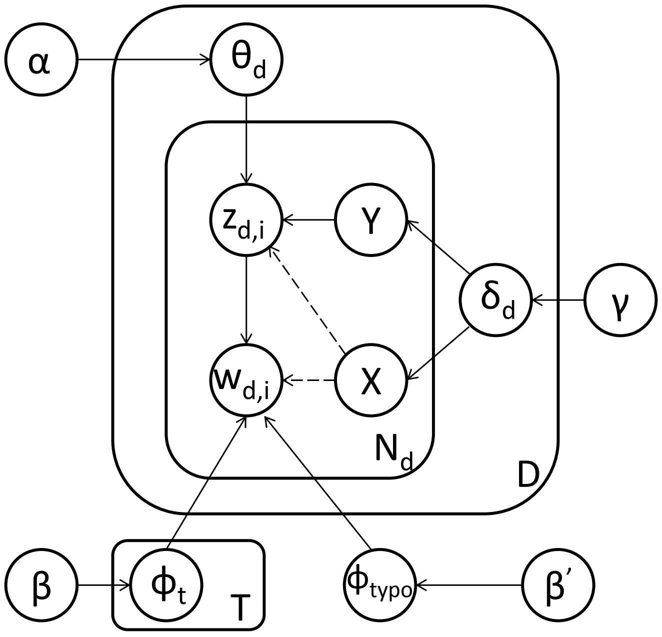


Figure 4.4. TDE-LDA Model.

the corpus. Without prior knowledge of grammar and syntax of human language or helps of dictionary, that is, machines cannot solely rely on the word frequency to tell the difference between a textual error and a rare word. To illustrate this point, we selected the `Reuters` newswire data set (to be explained in Section 4.4.1) and combined two OCR `Magazine` data sets. We calculated the percentages of words that appear from once to five times in the corpus. In Figure 4.5, the percentage curves of both typos and rare words exhibit very similar patterns in both corpora, making a computation-based differentiation hard. Therefore, our models adopt a supervised approach to distinguish rare words and textual errors in the document modeling process. One may use linguistic characteristics to differentiate typos in an unsupervised fashion. However, since the immediate goal of this work is to evaluate the validity of incorporating textual errors into document modeling process, we rather leave the development of more sophisticated modeling methods for future work.

Algorithm 3: **The TDE-LDA Model.**

```
1  For each of the $D$ documents $d$, sample $\theta_d \sim$ Dir($\alpha$)and $\delta_d \sim$ Beta($\gamma$);
2  For each of the $T$ topics $t$, sample $\phi_t \sim$ Dir($\beta$);
3  Sample $\phi_{typo} \sim$ Dir($\beta'$);
4  foreach $N_d$ words $w_{d,i}$ in document $d$ do
5      Sample a flag $X \sim$ Binomial($\delta_d$);
6      if $X = 1$ then
7          Sample a flag $Y \sim$ Binomial($\delta_d$);
8          if $Y = 1$ then
9              Assign a topic $z_{d,i} = z_{d,i-1}$;
10         endif
11         if $Y = 0$ then
12             Sample a topic $z_{d,i} \sim$ Multi($\theta_d$);
13         endif
14         Sample a word $w_{d,i} \sim$ Multi($\phi_{z_{d,i}}$);
15     endif
16     if $X = 0$ then
17         Sample a word $w_{d,i} \sim$ Multi($\phi_{typo}$);
18     endif
19 endfch
```

Topic vs. Term Dependency

The bigram topic model and n-grams model we mentioned in section 4.3.2 determine whether to form a bigram or an n-gram based on the surrounding words in the document. Although these models show better generalization performance over LDA, we argue that incorporating term dependency is not suitable in noisy text data for two reasons. First, in noisy document corpora, simply forming bigram or n-gram between consecutive words will increase the overall error rate. This is because an erroneous word will impact both the previous word and the succeeding word in terms of term combination. But it only impacts itself under the bag-of-words assumption for documents. Secondly, even though our TE-LDA model has a mechanism to distinguish between textual errors and correct words, by skipping typos the document model may generate incorrect bigram or n-gram

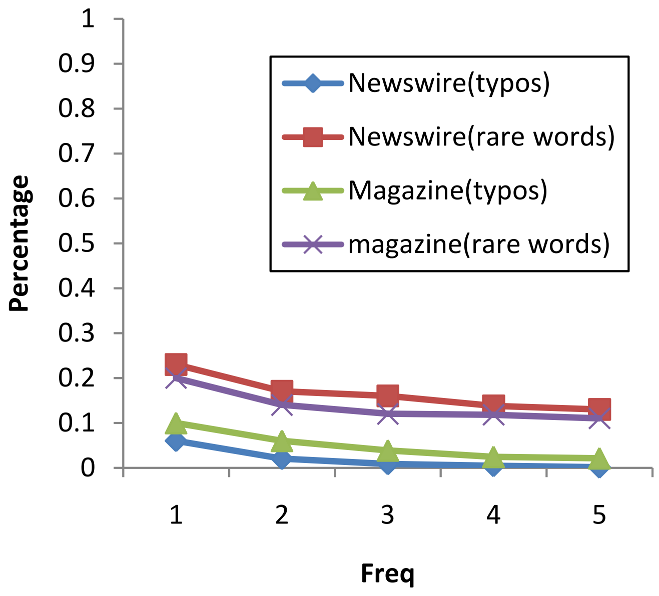


Figure 4.5. Comparison of percentages of typos and rare words.

word combinations which, in turn, decreases the accuracy of generalization performance. Therefore, we only consider topic dependency in order to capture the semantic relation of words. As a result, in this chapter, we select the traditional LDA model and the n-grams model without error modeling as baselines.

4.4 Experimental Validation

In order to validate our proposed models, we applied it to the *document modeling* problem. We trained our new models as well as the traditional LDA model on both synthetic and real text corpora to compare the generalization performance of these models. The documents in the document corpora are treated as unlabeled and the goal is to achieve high likelihood on a held-out test data [13]. In our experiments, each model was trained on 90% of the documents in each data set with fixed parameters α=0.5, β=0.01, β'=0.01 and γ=0.1 for simplicity and performance.

Table 4.2. Summary of data sets.

Name	Error	Domain	# of documents	# of unique terms	AVG document length
`OCR Business`	real	business documents	220	4,556	252
`OCR Magazine`	real	magazine articles	320	9,842	462
`OCR Legal`	real	legal documents	300	4,608	339
`OCR Newspaper`	real	newspaper articles	240	5,948	346
`OCR Magazine2`	real	magazine documents	200	10,485	872
`OCR BYU`	real	communique documents	600	33,749	529
`TREC AP`	synthetic	newswire articles	16,333	23,075	458
`NIPS`	synthetic	proceedings	1,740	13,649	2,843
`Reuters`	synthetic	newswire articles	12,902	12,112	223

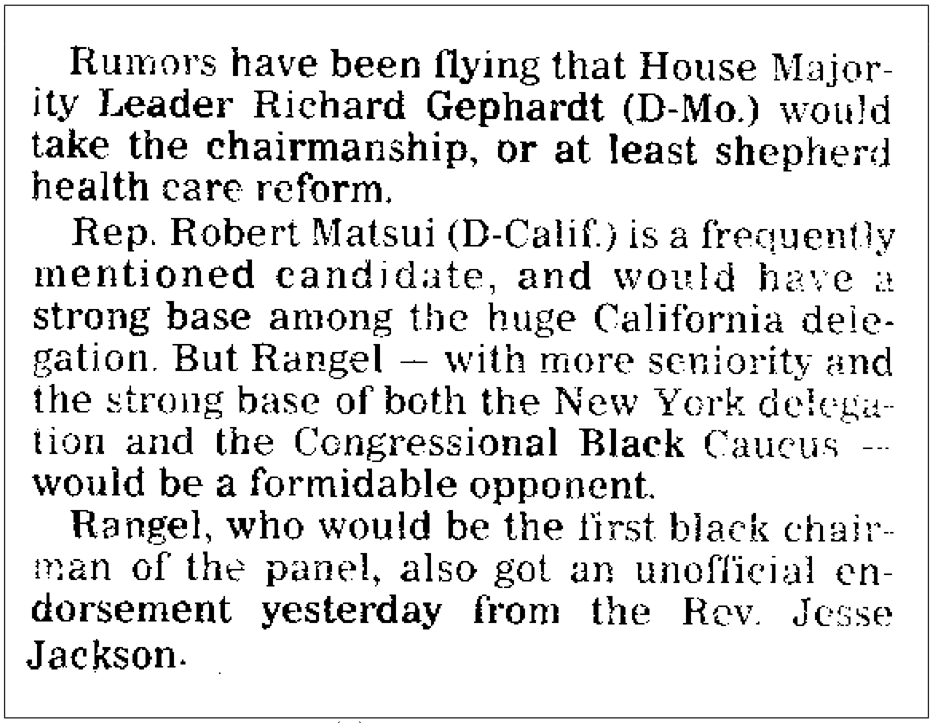
The trained model was used to calculate the estimate of the marginal log-likelihood of the remaining 10% of the documents.

4.4.1 Set-Up

Table 4.2 shows the summary of both real and synthetic data sets that we used in our experiments.

First, we prepared real data sets that contain varying degrees of errors in texts. From the PDF images in the data set, `Unlv`, using one of the most popular OCR engines (Google Tesseract), we converted PDF images to a textual document corpus. Since `Unlv` has the full texts as the ground truth, by comparing the transcript generated from OCR, we can exactly pinpoint which words are errors. In the end, we prepared five subsets: `Business`, `Magazine`, `Legal`, `Newspaper`, `Magazine2`. Similarly, we prepared another real corpus called `BYU`[1] which consists of 600 of the Eisenhower World War II communiques. This data set contains the daily progress of the Allied campaign until the German surrender. Example documents from `Newspaper` data set and `BYU` data set are shown in Figure 4.6. The quality of these originals is quite poor, hence the error rate is pretty high for the outputs of OCR engine. Note that in these real data sets, we cannot control the degrees of errors, and the error rates are determined by the OCR engine.

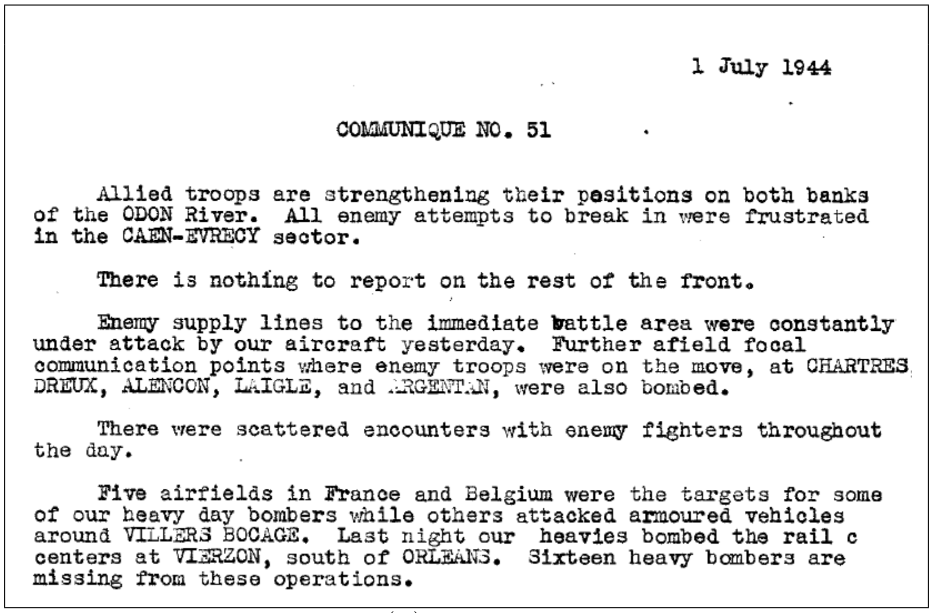
[1]`http://www.lib.byu.edu/dlib/spc/eisenhower`


Rumors have been flying that House Majority Leader Richard Gephardt (D-Mo.) would take the chairmanship, or at least shepherd health care reform.

Rep. Robert Matsui (D-Calif.) is a frequently mentioned candidate, and would have a strong base among the huge California delegation. But Rangel — with more seniority and the strong base of both the New York delegation and the Congressional Black Caucus — would be a formidable opponent.

Rangel, who would be the first black chairman of the panel, also got an unofficial endorsement yesterday from the Rev. Jesse Jackson.

(a) OCR Newspaper


1 July 1944

COMMUNIQUE NO. 51

Allied troops are strengthening their positions on both banks of the ODON River. All enemy attempts to break in were frustrated in the CAEN-EVRECY sector.

There is nothing to report on the rest of the front.

Enemy supply lines to the immediate battle area were constantly under attack by our aircraft yesterday. Further afield focal communication points where enemy troops were on the move, at CHARTRES DREUX, ALENCON, LAIGLE, and ARGENTAN, were also bombed.

There were scattered encounters with enemy fighters throughout the day.

Five airfields in France and Belgium were the targets for some of our heavy day bombers while others attacked armoured vehicles around VILLERS BOCAGE. Last night our heavies bombed the rail c centers at VIERZON, south of ORLEANS. Sixteen heavy bombers are missing from these operations.

(b) OCR BYU

Figure 4.6. Example documents from UNLV and BYU data sets.

Second, to conduct more controlled experiments, we also prepared synthetic data sets. In particular, we used three well-known benchmark data sets in the document modeling literature: `TREC AP`, `NIPS`, and `Reuters-21578`. The TREC Associate Press (AP) data set[2] contains 16,333 newswire articles with 23,075 unique terms. The `NIPS` data set[3] consists of the full text of the 13 years of proceedings from 1988 to 2000 Neural Information Processing Systems (NIPS) Conferences. The data set contains 1,740 research papers with 13,649 unique terms. The `Reuters-21578` data set[4] consists of newswire articles classified by topics and ordered by their date of issue. The data set contains 12,902 documents and 12,112 unique terms.

For all the above synthetic data sets, we generated erroneous corpora to simulate different levels of *Word Error Rates* (WER) – i.e., the ratio of word insertion, substitution and deletion errors in a transcript to the total number of words. Then, we closely studied the impact of textual errors in document modeling. In our experiments, we used three types of edit operations (i.e., insertion, deletion and substitution) in all the documents as follows: (1) insertion: a number of terms are randomly selected in a uniform fashion to insert a single character into the terms; (2) deletion: a number of terms are randomly selected in a uniform fashion to delete a single character from the terms; (3) substitution: a number of terms are randomly selected in a uniform fashion to change a single character of the terms. Note that multiple edit operations are not allowed for a single word. Let S, D and I denote the number of substitution, deletion and insertion operations, and let N denote the total number of words. Then, WER is calculated as follows. The procedure repeats until the desirable WER is achieved.

[2]`http://www.daviddlewis.com/resources/testcollections/trecap/`
[3]`http://www.cs.nyu.edu/~roweis/data.html`
[4]`http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html`

$$WER = \frac{S + D + I}{N}$$

4.4.2 Evaluation Metrics

The purpose of document modeling is to estimate the density distribution of the underlying structure of data. The common approach to achieve this goal is by evaluating the document model's generalization performance on new unseen documents. In our experiments, we calculated the *perplexity* of a held-out test set to evaluate the models. In language modeling, the perplexity quantifies the goodness of measuring the likelihood of a held-out test data to be generated from the learned distribution of the trained model. In particular, it is monotonically decreasing in the likelihood of the test data, which means a lower perplexity score corresponds to better generalization performance of the document model. Formally, for a test data of D_{test} documents the perplexity score is calculated as follows [13, 82]:

$$perplexity(D_{test}) = \exp\{\frac{-\sum_{d=1}^{D_{test}} \log p(\mathbf{w}_d)}{\sum_{d=1}^{D_{test}} N_d}\}$$

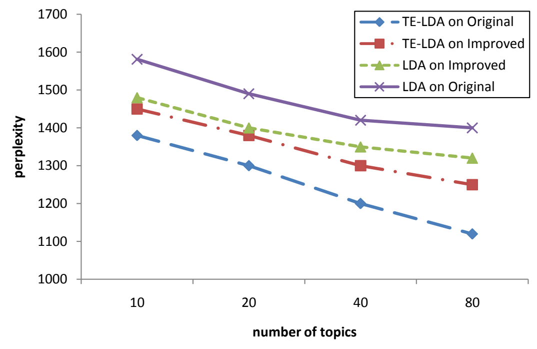
$$p(\mathbf{w}_d) = \sum_{k=1}^{K} p(w_d|z_k) p_{test}(z_k|d)$$

In the above equations, the probability $p(w_d|z_k)$ is learned from the training process and $p_{test}(z_k|d)$ is estimated through an additional Gibbs sampling process on the test data based on the parameters ϕ and δ learned from training data.

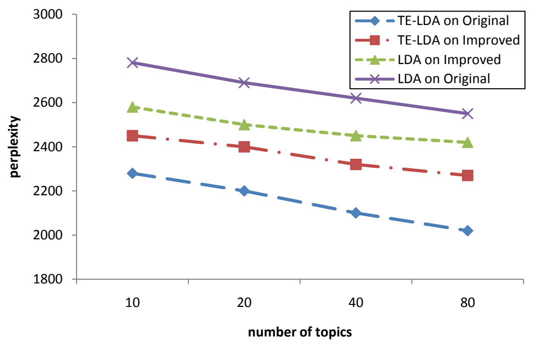
4.4.3 Comparison between TE-LDA and Baseline LDA

We first examine the performance of our TE-LDA model on real OCR data sets. Note that our immediate objective is to evaluate the validity of incorporating textual errors into document modeling process. This is based on the fact that most large-scale text data are machine-generated and thus inevitably contain many types of noise. As a novel solution, our TE-LDA model is developed from the traditional LDA model by adding a switch variable into the term generation process in order to tackle the issue of noisy text data. Hence, in this experiment, we compare the generalization of our TE-LDA model with the traditional LDA on various erroneous OCR text data. For example, each subset of real OCR data `Unlv` has a fixed WER, determined by the OCR engine. Due to the poor quality of PDF images and imperfect OCR process, WERs range from 0.2856 to 0.3739. That is, about 28–37% of words in the corpus could be erroneous words. Similarly, the WER of real OCR data `BYU` is as high as 48%.

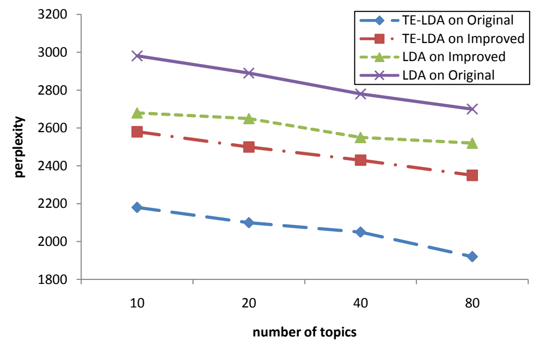
Recently, [3] proposed an algorithm for applying topic modeling to OCR error correction. The algorithm builds two models on an OCR document. One is a topic model which provides information about word probability and the other is an OCR model which provides the probability of character errors. The algorithm can reduce OCR errors by around 7%. We use the same error detecting technique to further correct our six real OCR data sets and then compare the performance of our TE-LDA model with the traditional LDA again. By doing so, we aim at finding out how the behavior of both topic models changes as the error rate changes on real OCR data. Figure 4.7 shows the perplexity of TE-LDA and LDA as a function of the number of hidden topics (e.g., 10, 20, 40, and 80). As we can see, our proposed TE-LDA model consistently outperforms the traditional LDA model on



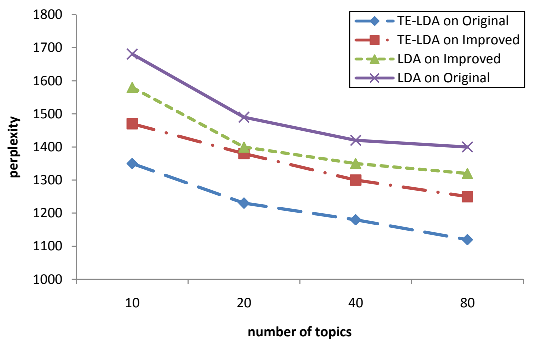
(a) `Business`



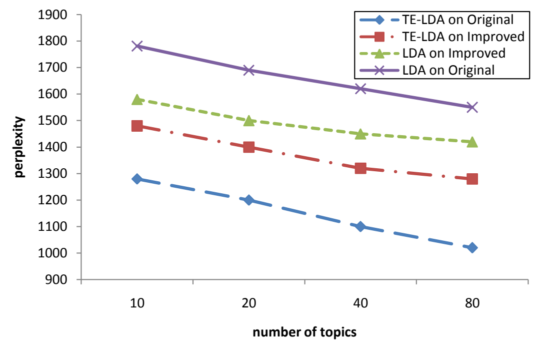
(b) `Magazine`



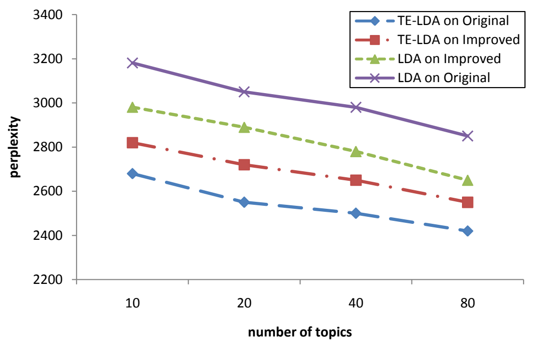
(c) `Legal`



(d) `Newspaper`



(e) `Magazine2`



(f) `BYU`

Figure 4.7. Perplexity of different models in original and improved `Unlv` and `BYU` data sets. From (a) to (f), data sets are `Business`, `Magazine`, `Legal`, `Newspaper`, `Magazine2` from `Unlv` and `BYU`. The WER of original data sets are 0.2856, 0.3158, 0.3148, 0.372, 0.3739 and 0.4856, respectively. The WER of improved data sets (using the technique from [3]) are 0.2653, 0.2893, 0.2985, 0.3468, 0.3518 and 0.4438, respectively.

both original and improved `Unlv` as well as `BYU` data sets. An interesting finding is that LDA performs better on improved corpora while TE-LDA performs better on original corpora. This is reasonable because our model is specifically designed to deal with textual errors in modeling noisy text documents and can achieve better generalization performance as the word error rates increase.

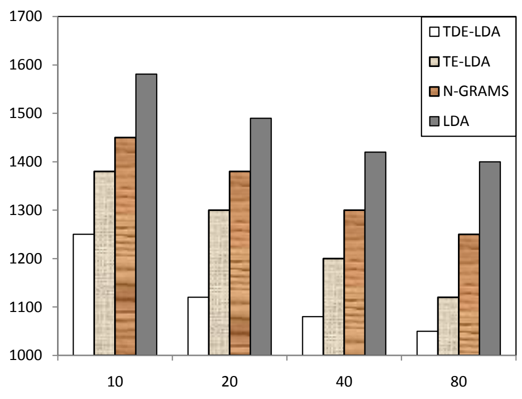
4.4.4 Comparison among Different Models

In this section, we systematically evaluate the performance of different models using various real and synthetic data sets. Since our purpose is to understand the performance of document modeling in erroneous environment, we compare the performance of our proposed models and the baseline models without removal of typos in text corpora.

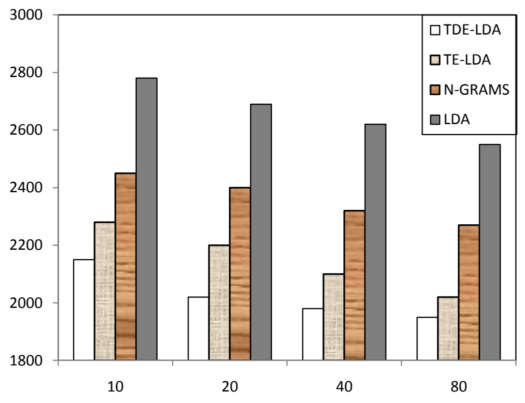
Results using Real Data Sets

We first compare the performance of our proposed models with the traditional LDA model and Wang's n-grams model on the real OCR data sets. Figure 4.8 shows the perplexity of TE-LDA and TDE-LDA models as a function of the number of hidden topics (e.g., 10, 20, 40, and 80) on the five subsets of `Unlv` corpus and the `BYU` corpus. As we can see, despite high WERs and different document themes among these data sets, our proposed TE-LDA and TDE-LDA models consistently outperform the traditional LDA model and the n-grams model. Note also that TDE-LDA is the best among the proposed models and the baseline models, which demonstrates that considering topic dependency improves the generalization performance of topic models in the context of noisy data.

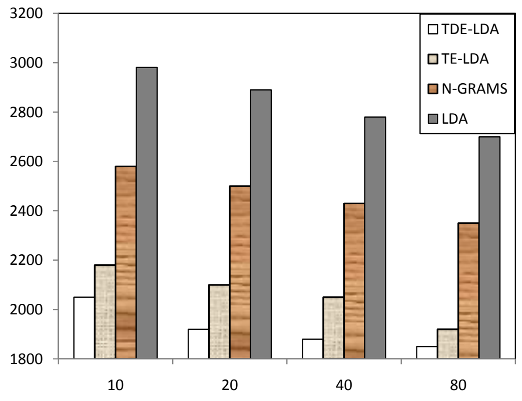
Table 4.3 shows examples of top words selected by LDA and the n-grams model as well as our models on the topic 3 of Table 1.1. From the table, note that



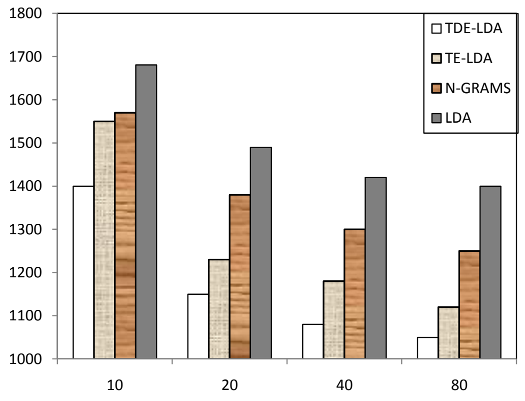
(a) Business



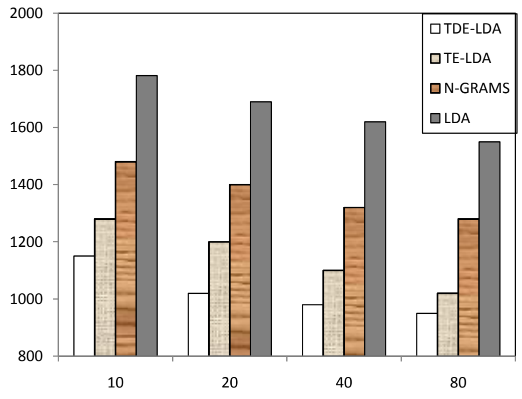
(b) Magazine



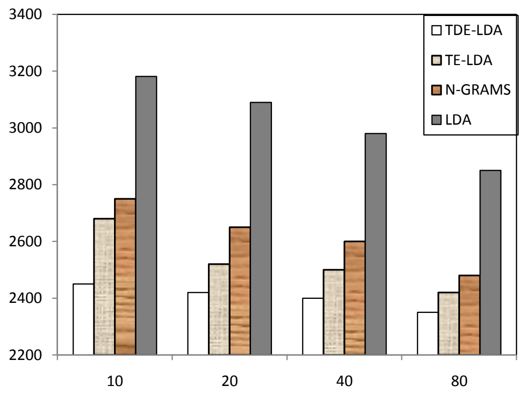
(c) Legal



(d) Newspaper



(e) Magazine2



(f) BYU

Figure 4.8. Perplexity of different models as a function of the number of topics (X-axis) in Unlv and BYU data sets. From (a) to (f), the data sets are Business, Magazine, Legal, Newspaper, Magazine2 from Unlv and BYU. The fixed word error rates (WER) of these data sets are 0.2856, 0.3158, 0.3148, 0.372, 0.3739 and 0.4856, respectively. Note the relatively high WERs due to the poor quality of PDF images in Unlv and BYU data sets.

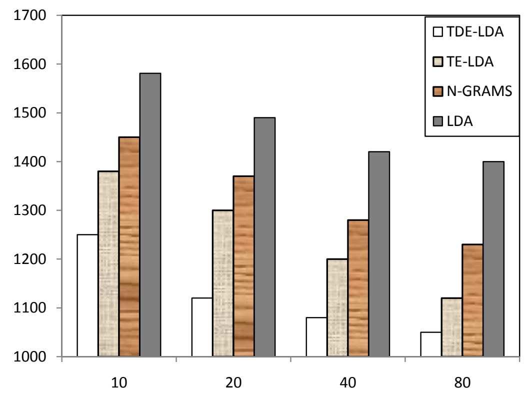
Table 4.3. Comparison of the selected top words using LDA vs. N-grams vs. our proposed models on a small sample of `Unlv` OCR data set. OCR-introduced erroneous words are in *italic*.

Model	Top words
LDA	air, *airlin*, *fli*, american, *engin*, subject, *threate*
N-GRAMS	american *airlin*, air flight, *threate* flight, *boe* plane
TE-LDA	air, american, plane, flight, bomb, pilot, airport
TDE-LDA	air, plane, pilot, american, passenger, aboard, bomb

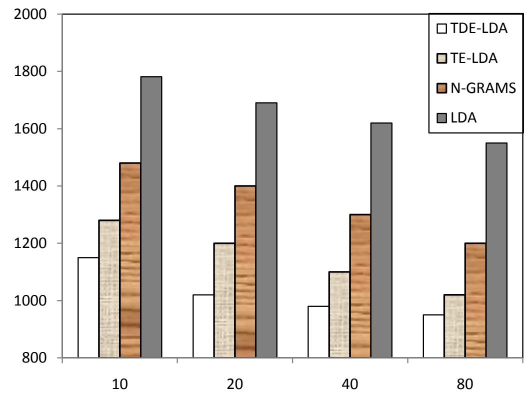
LDA suffers from choosing many OCR-introduced erroneous words as top words. Furthermore, the n-grams model tends to select several erroneous n-gram words as well. On the contrary, both TE-LDA and TDE-LDA models selected no erroneous top words, highlighting the superiority of our models in dealing with noisy text data. Overall, compared to others, our TDE-LDA model can select meaningful and generic top words or highly related words and make the topic more understandable.

Results using Synthetic Data Sets

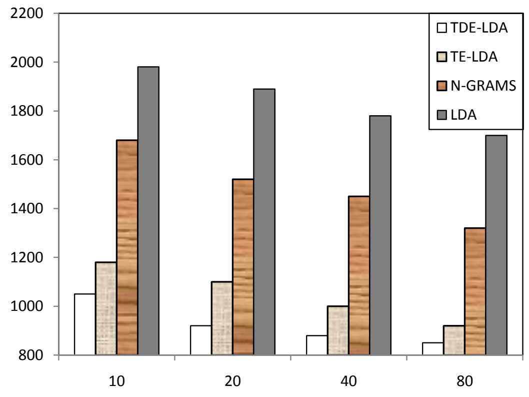
We then systematically compare the performance of our proposed models with the traditional LDA model as well as Wang's n-grams model on the synthetically generated erroneous corpora. In this comparison, we simulate different levels of WER (e.g., 0.01, 0.05, 0.1). Figures 4.9(a)-(c) show the perplexity of TE-LDA and TDE-LDA models as a function of the number of hidden topics in the `TREC AP` corpus. As we can see from Figures 4.9(a)-(c), at different levels of WER, our TE-LDA and TDE-LDA models consistently outperform the traditional LDA model. Furthermore, as WER increases, the margin of improvement increases. This is due to the incorporation of textual errors into the generation of terms in the document modeling process. We can also see that the models with consideration of topic or term dependency outperform the ones without that, regardless of whether we take into account textual errors during term generation. However, TDE-LDA is



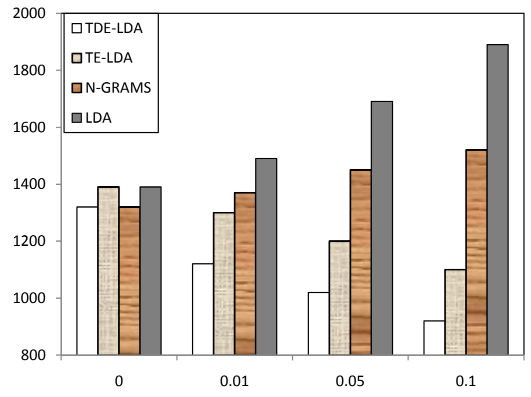
(a) WER = 0.01



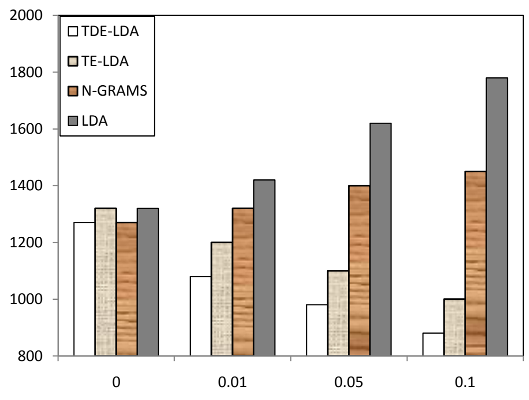
(b) WER = 0.05



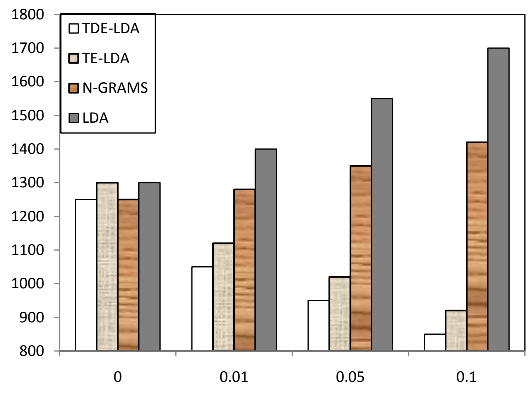
(c) WER = 0.1



(d) K = 20

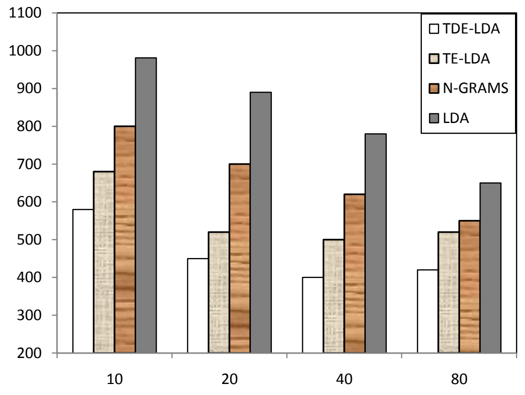


(e) K = 40

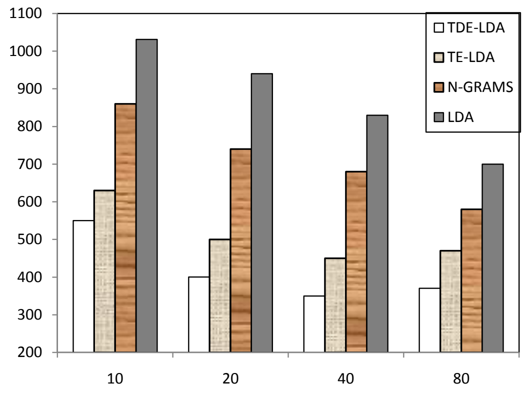


(f) K = 80

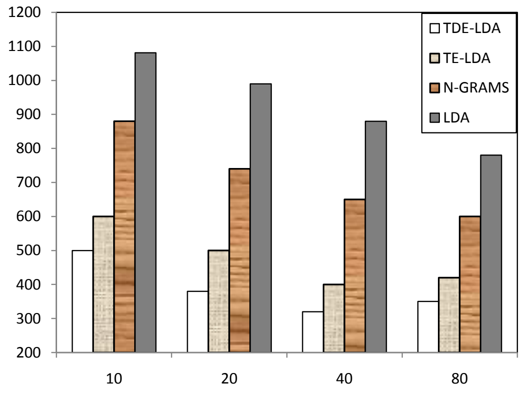
Figure 4.9. Performance summary using `TREC AP` data set. Perplexity of different models as a function of the number of topics (X-axis) in (a)-(c). Perplexity of different models as a function of WER (X-axis) in (d)-(f).



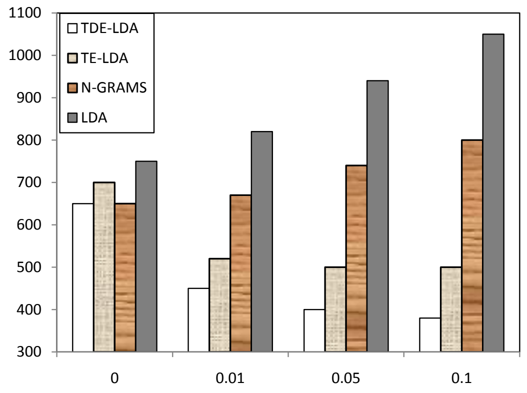
(a) WER = 0.01



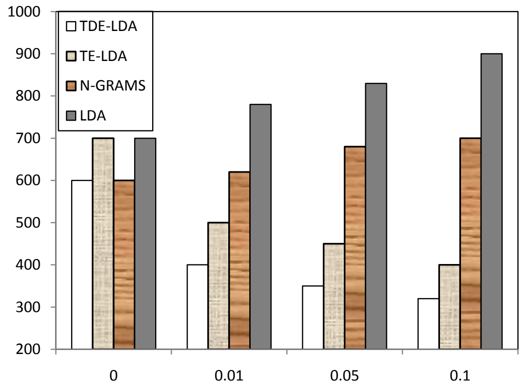
(b) WER = 0.05



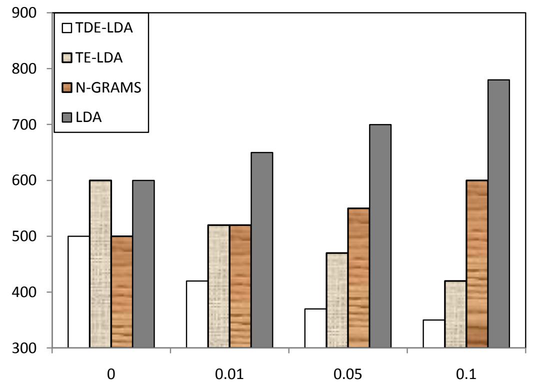
(c) WER = 0.1



(d) K = 20

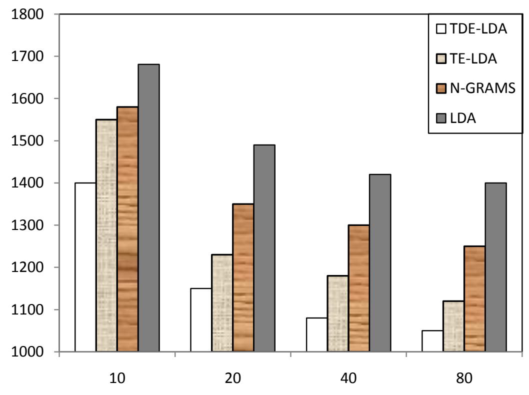


(e) K = 40

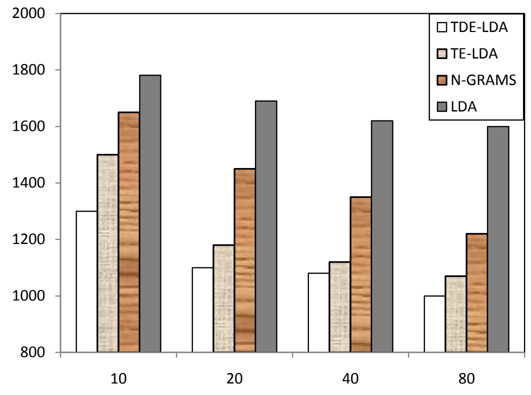


(f) K = 80

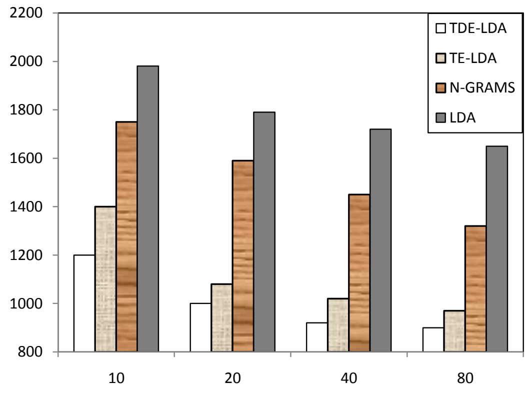
Figure 4.10. Performance summary using `NIPS` data set. Perplexity of different models as a function of the number of topics (X-axis) in (a)-(c). Perplexity of different models as a function of WER (X-axis) in (d)-(f).



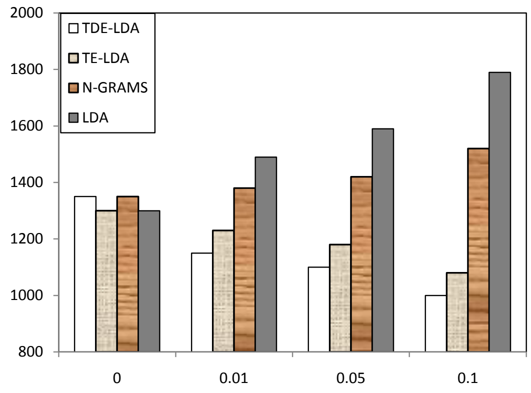
(a) WER = 0.01



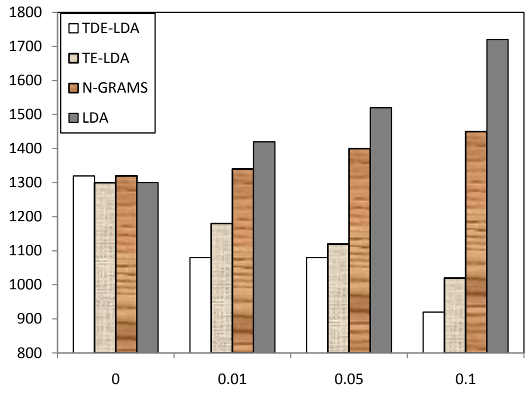
(b) WER = 0.05



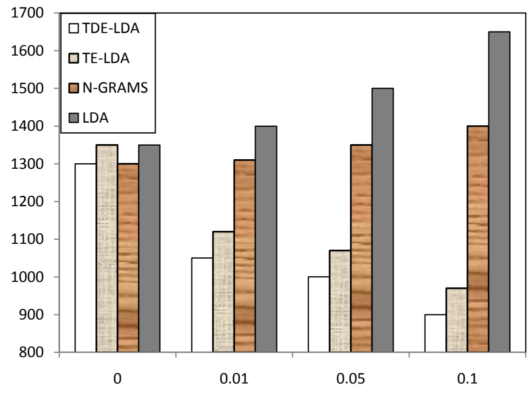
(c) WER = 0.1



(d) K = 20



(e) K = 40



(f) K = 80

Figure 4.11. Performance summary using `Reuters` data set. Perplexity of different models as a function of the number of topics (X-axis) in (a)-(c). Perplexity of different models as a function of WER (X-axis) in (d)-(f).

the best among the models and show better generalization of incorporating topic dependency in noisy text data. This demonstrates the improved performance of topic models with the removal of bag-of-words assumption.

In Figures 4.9(d)-(f), we fix the number of topics K and demonstrate how the different models perform as the WER increases in the `TREC AP` corpus. An interesting finding here is that the perplexity of both LDA and n-grams models increases as the word error rates increase. This is because these two models do not consider the errors in the term generation where the accuracy of calculation of word frequencies is affected. In contrast, our TE-LDA and TDE-LDA models outperform the other two and the margin of improvement increases as the word error rates increase. The experimental results in the `NIPS` (Figures 4.10) and `Reuters` (Figures 4.11) corpora show similar perplexity patterns.

4.5 Summary

In this chapter, we have proposed two extensions to the traditional LDA model to account for textual errors in latent document modeling. Our work is motivated by the facts that textual errors in document corpora are often abundant and separating words cannot completely capture the meaning of texts in data mining tasks. To overcome these constraints, we proposed our TE-LDA and TDE-LDA models to incorporate textual errors into the term generation process. Both TE-LDA and TDE-LDA adopt a switching mechanism to explicitly determine whether the current term is generated from the topic-document distribution through the general topic generation route or from a special word distribution through the typo processing route. However, TDE-LDA models the transition of topics between consecutive words as a first-order Markov process. Through extensive experiments, we have shown that our proposed models are able to model the document corpus in a more meaningful and realistic way, and achieve better generalization performance than the traditional LDA model and the n-grams model.

Chapter 5

Mining Social Content Using Time Series

5.1 Overview

Twitter, one of the most popular microblog sites, has been used as a rich source of real-time information sharing in everyday life. When Twitter users express their opinions about organizations, companies, brands, or sports in tweets, it in turn provides important opportunities for businesses in improving their services such as targeted advertising and personalized services. Since the majority of Twitter users' basic demographic information (e.g., gender, age, ethnicity) is unknown or incomplete, being able to accurately identify the hidden information about users becomes an important and practical problem. In this chapter, we study the problem of classifying Twitter users to a fixed set of categories based on the *content* of their tweets. Formally, we define our research problem as:

Definition (Twitter User Classification) *Given a set of Twitter users U, a stream of tweet messages $T_u = \{t_1, ..., t_{|T_u|}\}$ for each user $u \in U$, a pre-defined set* *of K class labels $C = \{c_1, ..., c_K\}$, and labeled samples such that $\langle u, c\rangle \in U \times C$, learn a classifier ψ: $U \rightarrow C$ to assign a class label to a unlabeled user.* □

The majority of existing solutions focused on using "textual" features of Twitter users (e.g., tweets messages) [25] or "network" features (e.g., follower/follwee network) [23] in classifying Twitter users. Despite their success, in this chapter, we argue that modeling tweet features as *time series* to amplify its *periodicity* pattern can be more effective in solving certain types of Twitter user classification problems.

In this chapter [84], we generate time series from tweets by exploiting the latent temporal information and solve the classification problem in time series domain. Our approach is inspired by the fact that Twitter users sometimes exhibit the *periodicity* pattern when they share their activities or express their opinions. We apply our proposed methods to both *binary* and *multi-class* classification of *sports* and *political* interests of Twitter users and compare the performance against eight conventional classification methods using textual features. Experimental results show that our *best* binary and multi-class approaches improve the classification accuracy over the *best* baseline binary and multi-class approaches by 15% and 142%, respectively.

5.2 Classifying User Interests using Textual Features

We first present the baseline classification approach for classifying users based on the *textual features* extracted from tweets. Given a stream of tweets, we represent each user as a document with a bag of words and directly extract features from the document content.

5.2.1 Feature Selection

We select two types of features based on tweet contents: TF-IDF and Topic Vector generated from Latent Dirichlet Allocation (LDA).

TF-IDF. Term Frequency – Inverse Document Frequency (TF-IDF) [79] is a classical term weighting method used in information retrieval. The idea is to find the important terms for the document within a corpus by assessing how often the word occurs within a document (TF) and how often in other documents (IDF). In our Twitter user setting, we have:

$$TF - IDF(t, u) = -\log \frac{df(t)}{U} \times tf(t, u)$$

where $tf(t; u)$ is the term frequency of word t within the stream of tweets of user u, $df(t)$ is the document frequency within the corpus (i.e., how many users' tweets contain at least one instance of t), and U is the number of users in the corpus.

Topic Vector. The Latent Dirichlet Allocation (LDA) proposed by [13] models documents by assuming that a document is composed by a mixture of hidden topics and that each topic is characterized by a probability distribution over words. This model provides a more compressed format to represent documents. In Twitter user classification, we adapt the original LDA by replacing documents with users' tweet streams. While LDA represents documents as bags of words, we represent Twitter users as words of their tweets. Therefore, a Twitter user is represented as a multinomial distribution over hidden topics. Given a number U of Twitter users and a number T of topics, each user u is represented by a multinomial distribution θ_u over topics, which is drawn from a Dirichlet prior with parameter α. A topic is represented by a multinomial distribution ϕ_t, which is drawn from another Dirichlet prior with parameter β. The topic vector acts as a low-dimensional feature representation of users' tweet streams and can be used as input into any classification algorithm. In other words, for each user u, we can use LDA to learn θ_u for that user and then treat θ as the features in order to do classification. The next step is to correctly assign a class label to each user in the reduced dimensional space.

5.2.2 Classification Methods

We select two popular classifiers over text domain: Naive Bayes (NB) and Support Vector Machines (SVM).

Naive Bayes. The Naive Bayes is a simple model which works well on text categorization [85], and it is a successful classifier based on the principle of Maximum A Posteriori (MAP). In this chapter, we adopt a multinomial Naive Bayes model. Given the user classification problem having K classes $\{c_1,\ c_2\ ...,c_K\}$ with prior probabilities $P(c_1)$,...,$P(c_K)$, we assign a class label c to a Twitter user u with feature vector $\mathbf{f} = (f_1, f_2..., f_N)$, such that

$$c = \arg\max_c P(c = c_k | f_1, f_2..., f_N)$$

That is to assign the class with the maximum a posterior probability given the observed data. This posterior probability can be formulated using Bayes theorem as follows:

$$P(c = c_k|f_1, f_2..., f_N) = \frac{P(c_k) \times \prod_{i=1}^{N} P(f_i|c_k)}{P(f_1, f_2..., f_N)}$$

where the objective is to assign a given user u having a feature vector $\mathbf{f}$ consisting of N features to the most probable class. $P(f_i|c_k)$ denotes the conditional probability of feature f_i found in tweet streams of user u given the class label c_k. Typically the denominator $P(f_1, f_2..., f_N)$ is not computed explicitly as it remains constant for all c_k. $P(c_k)$ and $P(f_i|c_k)$ are obtained through maximum likelihood estimates (MLE).

Support Vector Machines. The Support Vector Machines is another popular classification technique [86]. While Naive Bayes is a generative classifier to form a statistical model for each class, SVM is a large-margin classifier. The basic idea of applying SVM on classification is to find the maximum-margin hyperplane to separate among classes in the feature space. Given a corpus of U Twitter users and class labels for training $\{(\mathbf{f}_u, c_u)|u = 1, ..., U\}$, where $\mathbf{f}_u$ is the feature vector of user u and c_u is the target class label, SVM maps these input feature vectors into a high dimensional reproducing kernel Hilbert space, where a linear machine is constructed by minimizing a regularized functional. The linear machine takes the form of $\varphi(\mathbf{f}) = \langle \mathbf{w} \cdot \phi(\mathbf{f}) \rangle + b$ where $\phi(\cdot)$ is the mapping function, b is the bias and the dot product $\langle \phi(\mathbf{f}) \cdot \phi(\mathbf{f}') \rangle$ is also the kernel $K(\mathbf{f}, \mathbf{f}')$. The regularized functional is defined as:

$$R(\mathbf{w}, b) = C \cdot \sum_{u=1}^{U} \ell(c_u, \varphi(\mathbf{f}_u)) + \frac{1}{2}\|\mathbf{w}\|^2$$

where the regularization parameter $C > 0$, the norm of $\mathbf{w}$ is the stabilizer and

$\sum_{u=1}^{U} \ell(c_u, \varphi(\mathbf{f}_u))$ is empirical loss term. In standard SVM, the regularized functional can be minimized by solving a convex quadratic optimization problem which guarantees a unique global minimum solution.

5.3 Classifying User Interests using Time Series

In this section, we introduce our novel time series approach to tackle the problem of Twitter user classification. In particular, we propose a new technique for feature selection in order to convert Twitter users to time series by incorporating temporal information into the stream of tweets. We also propose two classification algorithms such that the multi-class Twitter user classification problem can be solved effectively in the time series domain.

5.3.1 Feature Selection

In this subsection, we explore the impact of temporal information in classifying Twitter users. Our assumption is that Twitter users often exhibit *periodicity* patterns when they post tweets to share their activities and statuses or express their opinions. This is because people from various categories tend to do different activities during different time frames. For example, sports fans usually post more relevant tweets about their favorite teams or players on game days during the season instead of offseason. Female shoppers love to share more of their opinions on Twitter during weekends or holidays. Travel enthusiasts tend to share more about their journey during summer time. [2] has shown that users participate in online social communities which share similar interests and there are recurring daily or weekly patterns in word usages. Another recent study [26] has also indicated that contents on microblogging platforms such as Twitter show patterns of temporal variation and pieces of content become popular and fade away in different temporal scales. Thus, we aim at leveraging *temporal* information in generating features from contents of tweet streams for our classification task. Our feature extraction process consists of two stages as follows.

Given a set of Twitter users U and K class labels $C = \{c_1,\ c_2\ ...,c_K\}$, first, we identify the *category-specific* keywords as a good source of relevant information of the entity in each class. In particular, we can harvest this kind of category related keywords from some external knowledge sources such as Wikipedia, or more directly, we can make use of online dictionaries such as WordNet, i.e. a network of words, to find all the related terms linked to the category keywords. For example, different sports have different Wikipedia pages consisting of rich corpora of sports-specific keywords which can be utilized to identify positive topics generated by sports fans in tweet streams. This keyword extraction process can be done manually or automatically depending on the scope of classification tasks and applications. The dictionary of these predefined keyword features serves as a rich representation of the entity in each category and contributes towards positive evidence of each class.

Second, given a stream of tweet messages $T_u = \{t_1,\ t_2,\ ...,t_{|T_u|}\}$ for each Twitter user u, we divide these tweets into segments based on predefined sliding time windows, e.g., daily or weekly time frames. We then record the number of word occurrences of category-specific keywords that appear in all the tweet messages within each sliding window. Based on these numbers, we convert each user into a numerical time series by calculating the frequency or percentage of keywords occurrences at different granularity levels, e.g., word or tweet levels. These time series reflect temporal fluctuations with respect to frequency changes of positive mentions of keywords in tweet messages from users in each class.

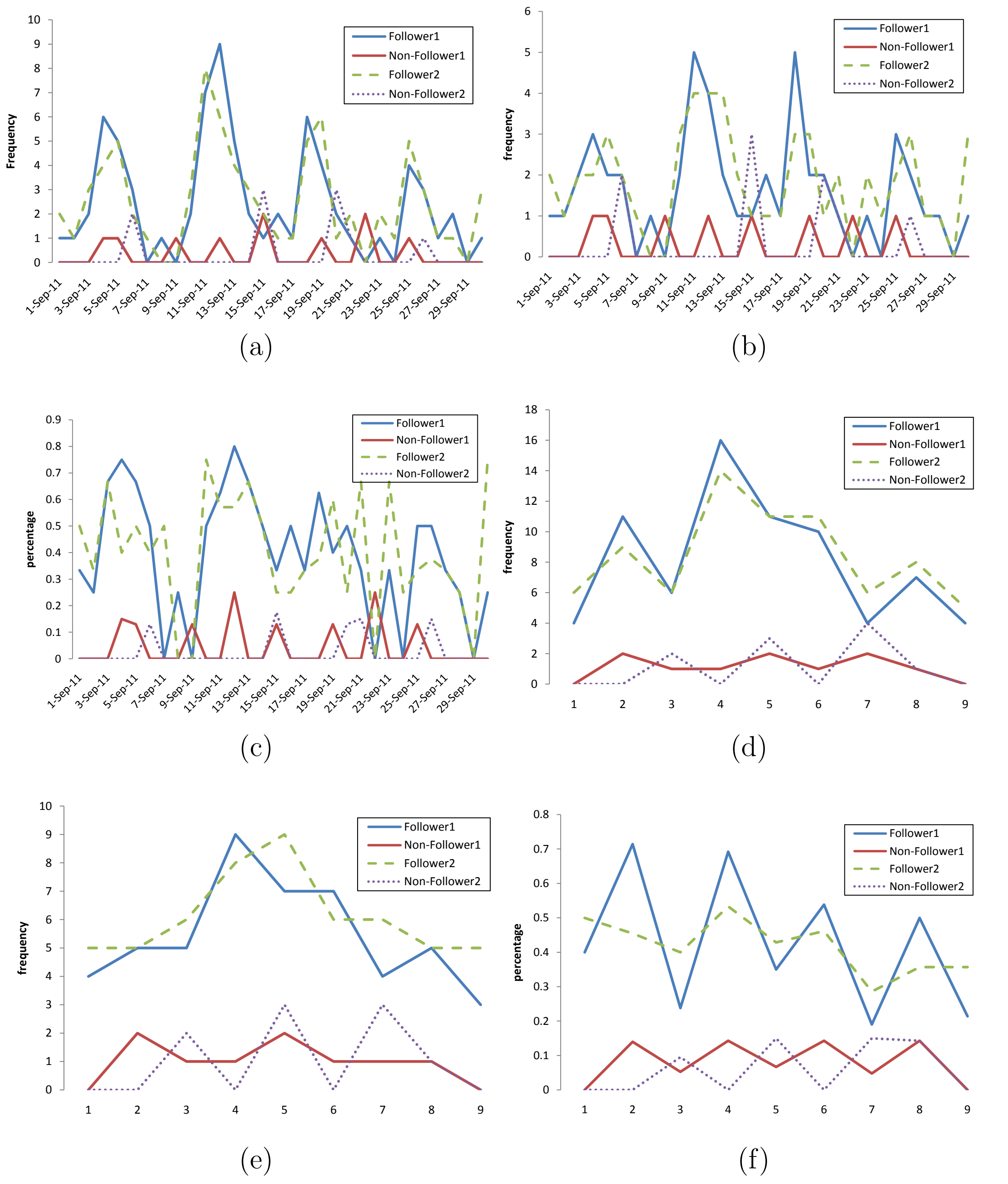


Figure 5.1. (a)-(c): daily time series based on the *frequency* of relevant *words*, *frequency* of relevant *tweets*, and *percentage* of relevant *tweets*; (d)-(f): weekly time series based on the *frequency* of relevant *words*, *frequency* of relevant *tweets*, and *percentage* of relevant *tweets*.

Example 5. As an illustration, consider Figure 5.1 that shows examples of using football-specific keywords (details shown in section 6.4.1) to generate daily and weekly time series of two followers and two non-followers of the NFL team, New York Giants, during the month of September 2011. Figures 5.1(a) and 5.1(d) show daily and weekly time series based on frequencies of football-specific words. On a daily basis, we treat a user's daily tweet streams as a bag of words and count the frequency of football-specific words that appear in the daily tweet messages. On a weekly basis, we treat a user's tweet streams on game days (i.e., Sunday and Monday) vs. non-game days (i.e., Tuesday through Saturday) separately. That is, we count the frequency of football-specific words that appear in tweet messages on game days vs. non-game days in each week. We can easily see that both daily and weekly time series of the followers preserve similar shapes in real-value domain (with some shifting) while the time series of the non-followers have rather different shapes. Figures 5.1(b) and 5.1(e) show daily and weekly time series based on frequencies of football-specific tweets. On a daily basis, we count the frequency of tweets containing football-specific words that appear in daily tweet messages. On a weekly basis, we count the frequency of tweets containing football-specific words that appear in the tweet messages on game days vs. non-game days separately. Figures 5.1(c) and 5.1(f) show daily and weekly time series based on the percentage of football-specific tweets (i.e., fraction of the number of tweets containing football-specific keywords over the number of tweet messages within each time frame). Note that regardless of particular feature extraction methods to generate time series, there is a clear difference between time series of football followers and non-followers. □

Each time series serves as a feature vector of the corresponding Twitter user

Algorithm 4: **Time Series Feature Extraction**.

```
Input  : A set of Twitter users $U$ and a stream of tweet messages $T_u = \{t_1, t_2, ..., t_{|T_u|}\}$ for each user $u$ with class label $c$ from a predefined set of $K$ class labels $C = \{c_1, c_2 ..., c_K\}$, a new user $v$ and a stream of tweet messages $T_v$
Output: A set of time series $TS = \{TS_1, TS_2, ..., TS_{|U|}\}$ where $TS_u$ is a converted time series feature vector for each user $u$
1  /*stage1: preprocessing*/;
2  for class in classList do
3      build category-specific keyword lists $list[class] = preProcess(class)$;
4  endfor
5  /*stage2: transformation*/;
6  for class in classList do
7      for $u$ in userList[class] do
8          break $T_u$ into smaller segments $S$;
9          for $s$ in $S$ do
10             count the number of occurrences $w_s$ of keywords from $list[class]$ in $s$ ;
11         endfor
12         convert user $u$ into a time series $TS_u$ from $w$; return($TS_u$);
13     endfor
14 endfor
```

for further classification in the domain of numerical signals. The detailed feature extraction process is shown in Algorithm 4.

5.3.2 Classification Methods

In time series classification, using feature-based methods as in section 5.2 is a challenging task because it is not easy to do feature enumeration on numerical time series data. Therefore, we use the common *distance-based* approach to classify time series. Previous research has shown that compared to commonly used classifiers such as SVM, k-nearest neighbor (kNN) classifier (especially 1NN) with dynamic time warping distance is usually superior in terms of classification accuracy [87].

kNN. The kNN is one of the simplest non-parametric classification algorithms, which does not need to pre-compute a classification model [88]. Given a labeled Twitter user set U, a positive integer k, and a new user u to be classified, the kNN classifier finds the k nearest neighbors of u in U, $kNN(u)$, and then returns the dominating class label in $kNN(u)$ as the label of user u. In particular, if $k = 1$, the 1NN classifier will return the class label of the nearest neighbor of user u in terms of distance in time series feature space.

5.3.3 Distance Functions

We select two types of distance functions for user classification in time series domain: Dynamic Time Warping (DTW) and Symbolic Aggregate approXimation (SAX).

DTW. The Dynamic Time Warping (DTW) is a well-known technique to find an optimal alignment between two time series [89]. Intuitively, the time series are warped in a nonlinear fashion to match each other. The idea of DTW is to align two time series in order to get the best distance by aligning. In data mining and information retrieval research, DTW has been successfully applied to automatically deal with time-dependent data. Given two Twitter Users' time series feature vectors $\mathbf{X} = (x_1, x_2..., x_{|X|})$ and $\mathbf{Y} = (y_1, y_2..., y_{|Y|})$, DTW is to construct a warping path $\mathbf{W} = (w_1, w_2..., w_K)$ with $\max(|X|, |Y|) \leq K < |X| + |Y|$ where K is the length of the warping path and w_k is the k^{th} element $(i, j)_k$ of the warping path. The optimal warping path is the path which minimizes the warping cost:

$$DTW(X, Y) = \min\{\sum_{k=1}^{K} d(w_k)\}$$

The optimal path can be found very efficiently using dynamic programming to evaluate the following recurrence:

$$\gamma(i,j) = d(i,j) + \min\{\gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1)\}$$

where $\gamma(i,j)$ denotes the cumulative distance as the distance $d(i,j)$ found in the current cell and the minimum of the cumulative distances of the adjacent elements.

SAX. The Symbolic Aggregate approXimation (SAX) is known to provide good dimension reduction and indexing with a lower-bounding distance measure [90]. In many data mining applications, SAX has been reported to be as good as well-known representations such as Discrete Wavelet Transform (DWT) and Discrete Fourier Transform (DFT). However, SAX requires less storage space. In this chapter, we adopt the same SAX technique in [90] for classifying Twitter users in time series domain.

5.3.4 Multi-class User Classification

We present two classification variations in time series domain for multi-class Twitter user classification.

One-Vs-All. The first approach is to reduce the problem of classifying among K classes into K binary problems and each problem discriminates a given class from the other $K-1$ classes. In this approach, we build K binary classifiers where the k^{th} classifier is trained with positive examples belonging to class k and negative examples belonging to the other $K-1$ classes. For the k^{th} binary classifier, we convert all users into time series using the category-specific keyword list of the k^{th} class. When classifying a new user v, the classifier with the nearest neighbor of

Algorithm 5: **One-Vs-All User Classification**.

```
Input  : A set of Twitter users $U$ and a stream of tweet messages $T_u = \{t_1, t_2, ..., t_{|T_u|}\}$ for each user $u$ with class label $c$ from a predefined set of $K$ class labels $C = \{c_1, c_2 ..., c_K\}$, a new user $v$ and a stream of tweet messages $T_v$
Output: The class label for user $v$
for class in classList do
    for u in userList do
        $TS_u$ = FeatureExtraction($u$);
    endfor
    $TS_v$ = FeatureExtraction($v$);
    /*classification*/;
    learn a $kNN$ classifier on time series $TS_v$ and $TS_u$ where $u \in U$;
endfor
/*pairwise comparison*/;
for class in classList do
    find the class with the best $kNN$ classifier ;
endfor
return(class);
```

the user is considered the winner, and the corresponding class label is assigned to the user v. The detailed classification algorithm is shown in Algorithm 5.

All-At-Once. The second approach is to convert Twitter users of each class c into time series simultaneously using the category-specific keyword list of the corresponding class. Given a new user v, we convert v using the combination of keyword lists of all classes. When classifying the new user v, the classifier returns the label of the nearest neighbor as the corresponding class label to be assigned to the user v. The detailed classification algorithm is shown in Algorithm 6.

5.4 Experiments on Sports Interests

In order to validate our classification approaches, we first apply them to both binary and multi-class classification problems with respect to identifying NFL foot-

Algorithm 6: **All-At-Once User Classification**.

```
Input  : A set of Twitter users $U$ and a stream of tweet messages $T_u = \{t_1, t_2, ..., t_{|T_u|}\}$ for each user $u$ with class label $c$ from a predefined set of $K$ class labels $C = \{c_1, c_2 ..., c_K\}$, a new user $v$ and a stream of tweet messages $T_v$
Output: The class label for user $v$
1  for class in classList do
2      for u in userList[class] do
3          $TS_u$ = FeatureExtraction($u$);
4      endfor
5      listAll += list[class]
6  endfor
7  convert user $v$ into a time series $TS_v$ using listAll ;
8  /*classification*/;
9  learn a $kNN$ classifier to find the best class on time series $TS_v$ and $TS_u$
   where $u \in U$;
10 return(class);
```

ball fans and team fans. Specifically, our experimental questions are the following:

1. *Binary*: How accurately can we predict if a Twitter user is a football fan or not?

2. *Multi-class*: How accurately can we predict the football team (1 out of 32 teams) of a Twitter user when she is known as a football fan?

5.4.1 Set-Up

Data Collection. We focused on the football season from Sep. 2011 to Dec. 2011 in the experiments. Starting from the 32 official Twitter accounts of NFL football teams, we first identified 1,000 followers per team (i.e., a total of 32,000 users) as the "fan" corpus. Similarly, we also identified a total of 32,000 users who do not follow any Twitter account of the football teams as the "non-fan" corpus. Each user has at least 3-4 tweets per day (i.e., about 400 tweets for 4-

month period). For each tweet, we removed the external links, non-alphabetic characters such as “@” and “#”, emoticons and stop words, and then filtered out tweets with less than five words. At the end, our data set included a total of 64,000 users and 2.56 million tweets. From the Wikipedia page of each of the 32 NFL teams, next, we automatically harvested the team and player names from the roster section, and manually identified football-specific keywords such as “nfl” and “quarterback” as well as team-specific keywords. This semi-automatic generation process of category-specific keywords resulted in a total of 2,330 unique terms at the end in our dictionary. Team-specific keywords serve as category-specific keywords for multi-class classification purpose while the combination of team-specific and football-specific keywords serve as category-specific keywords for binary classification purpose.

Evaluation Metrics. The binary classification task is to classify the users into two classes, i.e., one class which represents the users who are fans of NFL football (positive class) and the other class which represents user that are not fans of NFL football (negative class). Moreover, the multi-class classification task is to classify the users into 32 classes with each representing the fans of each individual NFL team. For evaluation purpose, all the users can be grouped into four categories, i.e., true positive (TP), true negatives (TN), false positives (FP) and false negatives (FN). For example, the true positives are the users that belong to positive class and are in fact classified to the positive class, and the false positives are the users not belonging to positive class but incorrectly classified to the positive class. Since we are interested in both positive and negative classes especially in multi-class classification, we use the *accuracy (ACC)* metric to measure the performance of our different classifiers as follows:

$$ACC = \frac{TP + TN}{TP + FP + TN + FN}$$

In all subsequent experiments, we use the *10-fold cross validation* [25] to measure the accuracy.

Baseline Method. We use two types of baselines. First, the naive keyword-based (KB) classification uses the category-specific keywords when classifying a Twitter user. Given a stream of tweets from a user, we count the percentage of keywords from each category-specific keyword list present in the tweet corpus. If the percentage exceeds a predefined threshold, then the user is classified into the positive class. If there is a tie, then the class label with higher percentage ratio is returned. Second, the NB or SVM based classification using the textual features in Section 5.2 serves as the more sophisticated baseline. Finally, we compare the accuracy of our proposed time-series based classification against these two types of baselines.

5.4.2 Binary Classification

Given a stream of tweets from a user, the goal of binary classification is to predict whether the user is likely to follow NFL football teams, i.e., whether the user is a football fan or not. In this task, we combine all the tweets crawled for each of the 32 NFL teams and their fans as positive examples (i.e., 32,000 positive users) and similarly combine all the tweets from the users who do not follow any of the teams as negative examples (i.e., 32,000 negative users).

As to the baselines, we used a total of 8 approaches, all of which use features in *text* domain. First, we tested two approaches using the keyword-based classifier at word level (Word+KB) and tweet level (Tweet+KB). The Word+KB (resp. Tweet+KB) computes the percentage of words (resp. tweets) containing football-related keywords and uses a simple threshold (e.g., 10%) to classify a user into the positive class. Second, we prepared 6 baselines using three variations of features (i.e., TF-IDF and LDA with 20 and 100 topics) and two classifiers (i.e., NB and SVM).

As to our proposed methods, we first used football-specific keywords to convert each user's tweets into a time series on both daily and weekly time scales. On a daily scale, we treat a user's daily tweet streams as a bag of words and count the number of football-specific words (DW) or football-specific tweets (DT) that appear in the daily tweet messages. On a weekly scale, we treat a user's tweet streams on game days (i.e., Sunday and Monday) and non-game days (i.e., Tuesday through Saturday) separately within each week. Then, we prepared two variations – weekly words (WW) and weekly tweets (WT). Next, we used two distance functions (i.e., DTW and SAX) and kNN classifier to do classification in time series domain. Previous research has shown that compared to commonly used classifiers such as SVM, 1-nearest neighbor (1NN) classifier with the DTW distance usually yields superior classification accuracy [87]. Therefore, in this chapter, we applied 1NN classifier for simplicity purpose.

Figure 5.2 shows the performance comparison of two types of baseline approaches. We can clearly see that TF-IDF or LDA based methods show much improvements over the keyword-based baseline. First, we can observe that keyword-based baseline at tweet level slightly outperforms the word-level baseline. This is reasonable because as long as a user's tweet contains a category-specific keyword, the classifier treats the entire tweet relevant to the positive class and this in turn

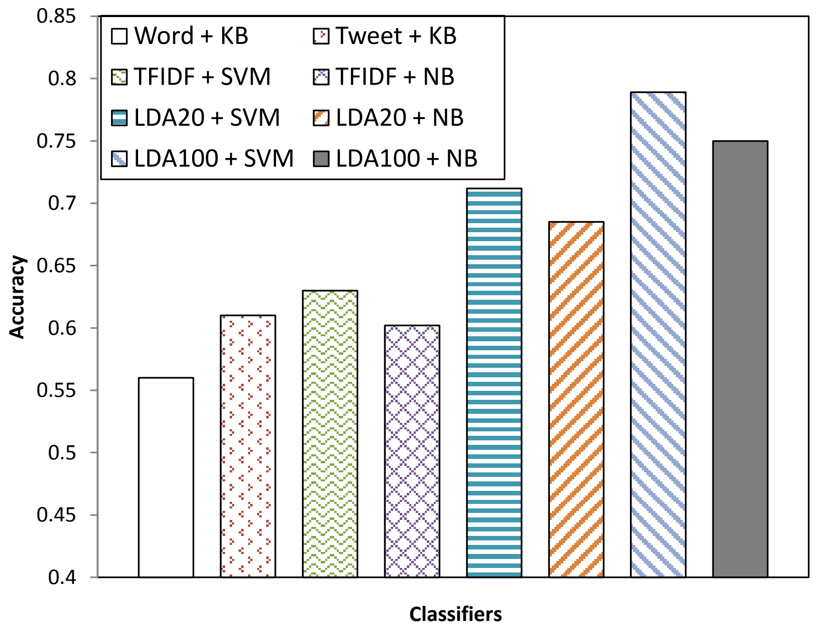


Figure 5.2. Binary classification of Twitter users in text domain.

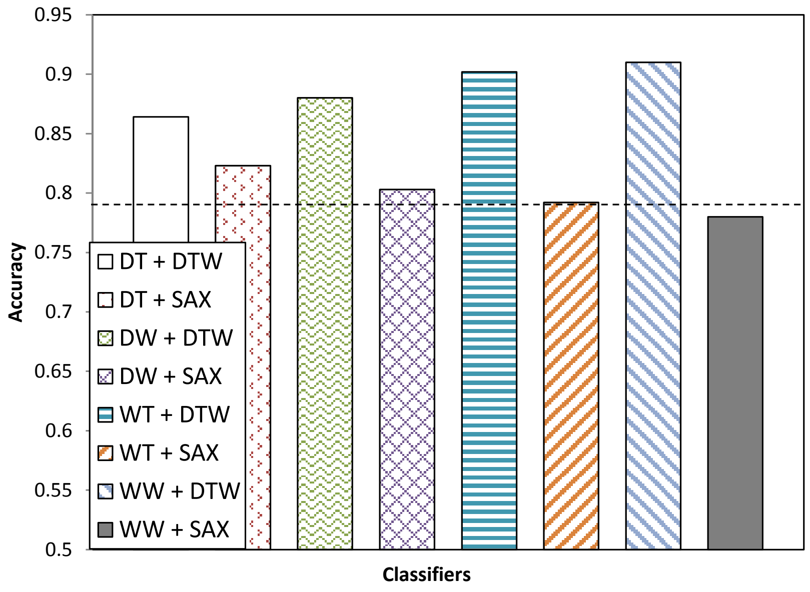


Figure 5.3. Binary classification of Twitter users in Time Series domain. (Note that DT, DW, WT, WW are daily time series at tweet level, daily time series at word level, weekly time series at tweet level, weekly time series at wold level, respectively. Dotted line denotes the accuracy of the “best” classifier in text domain from Figure 5.2.)

increases the *relatedness* of the user's tweet stream to the positive class. Second, regarding difference between features extracted from tweet contents, we can see that classifiers using the topic feature derived from topic models outperform classifiers using the TF-IDF feature. For example, SVM classifier using topic feature outperforms SVM classifier using TF-IDF and improves the classification accuracy by up to 25%. This is consistent with [24] as topic-based linguistic features are consistently reliable and more discriminative in user classification tasks. Third, using either TF-IDF feature or topic feature, SVM classifier generally outperforms NB classifier. This is also consistent with previous experimental results which show that SVM performs better than NB in general classification tasks [86]. And finally, Figure 5.4 shows the impact of number of topics to the Naive Bayes and SVM classifiers using topic feature. We can see that as number of topics increases, the accuracy of the corresponding classifiers also increases and converges at around 100 topics.

Figure 5.3 shows the performance of our proposed time series approach for user classification. First, we can see that our 1NN classifier using DTW or SAX as distance functions generally performs better than *all* baseline methods in Figure 5.2. For example, 1NN classifier using time series feature on a weekly basis and DTW as distance function outperforms SVM classifier using topic feature by improving the classification accuracy by around 15%, and outperforms NB classifier using topic feature by 22%. Second, regarding user classification in time series domain, 1NN classifier using DTW as distance function generally outperforms 1NN classifier using SAX as distance function. This is due to the fact that SAX is actually used as a symbolization technique for dimension reduction specifically in time series classification. Our time series approach consists of a transformation process to convert textual features to time series features, thus further symbolizing the time

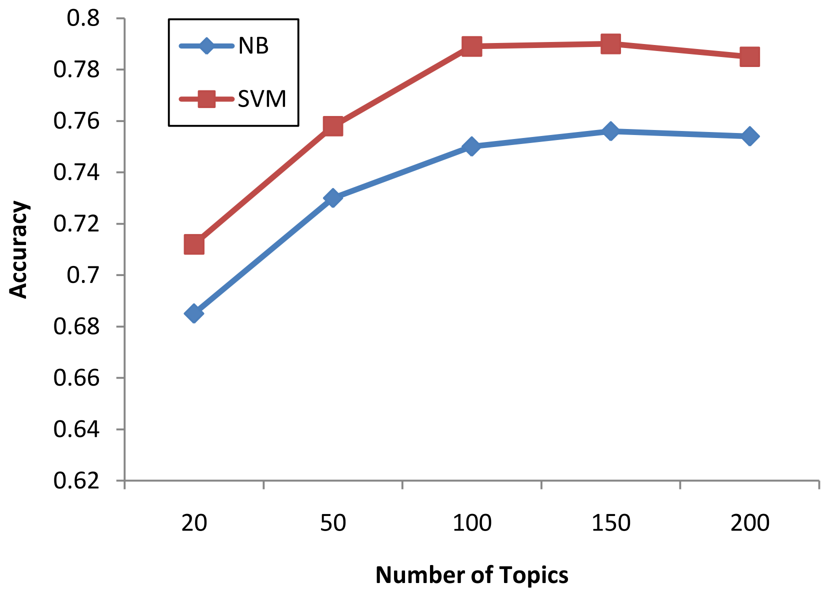


Figure 5.4. Impact of the number of topics in LDA topic features.

series may not be necessary and consequently results in some loss of information. However, the performance of 1NN classifier using SAX is still comparable to or slightly better than the performance of SVM classifier using topic feature.

5.4.3 Multi-class Classification

Next, the goal of multi-class classification is to predict which particular team (out of 32 NFL football teams) a given user is a fan of. In this task, we used the corpus with a total of 32,000 fans, i.e., 1,000 users per class. Similar to Section 5.4.2, we applied 8 baselines using features in text domain and 8 variations of our proposal in time series domain. In addition, we adopted two alternatives to evaluate multi-class classification scenario, as illustrated in Algorithms 5 and 6.

Figure 5.5 compares the multi-class classification accuracy among 8 baseline methods in text domain. Again, NB or SVM based baseline methods outperform keyword-based heuristics. First, regarding different features extracted from tweet

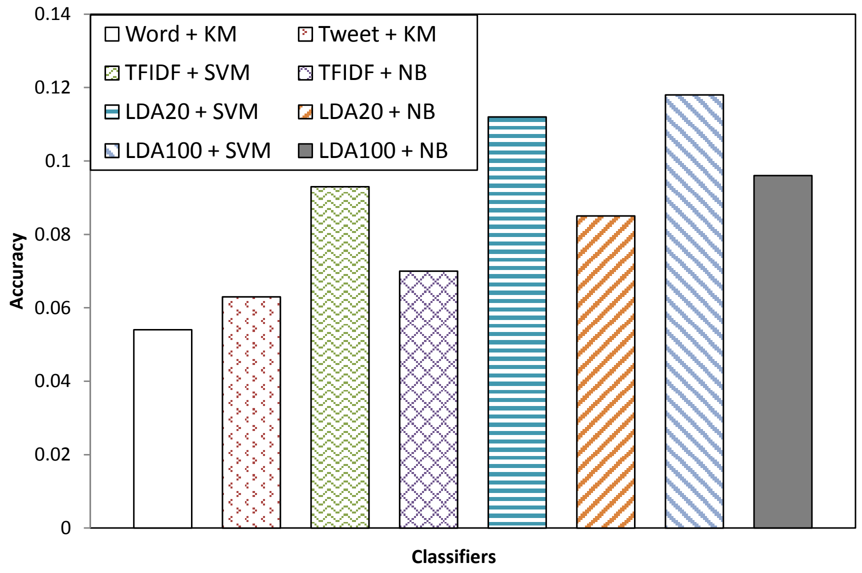


Figure 5.5. Multi-Class classification of Twitter users in text domain.

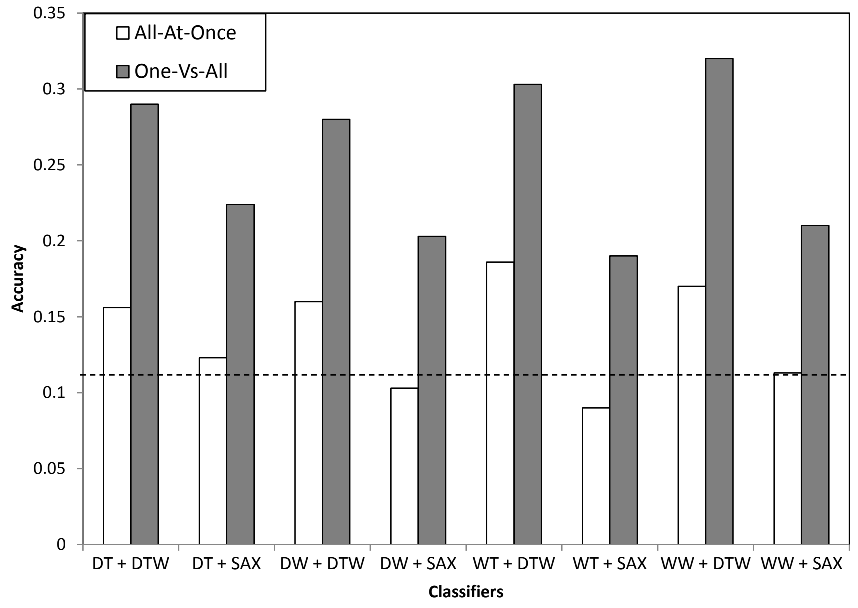


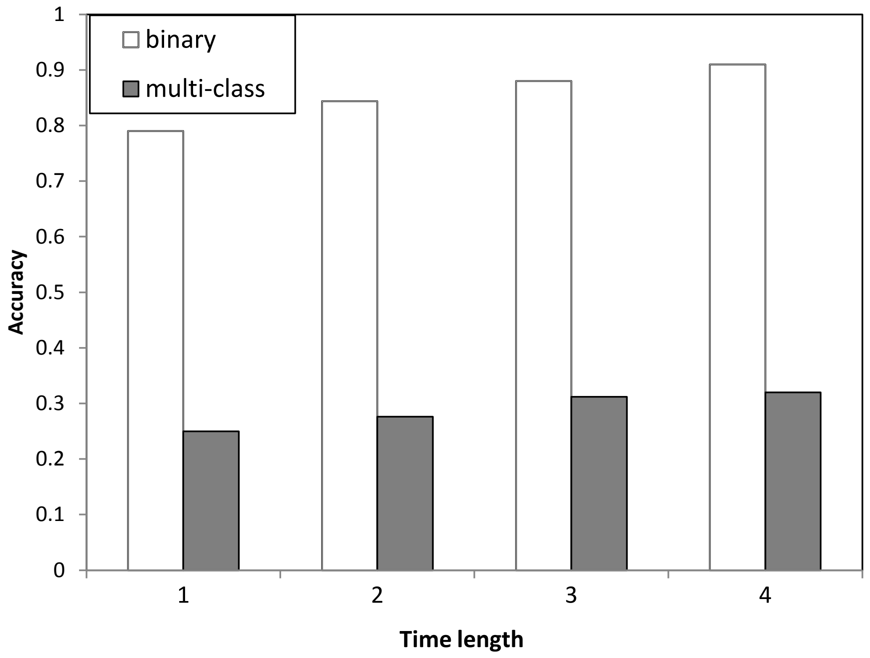
Figure 5.6. Multi-Class classification of Twitter users in Time Series domain. (Refer to Figure 5.3 for notations. Dotted line denotes the accuracy of the "best" multi-class classifier in text domain from Figure 5.5.)

contents, it is shown that classifiers using the topic feature derived from topic models outperform classifiers using the TF-IDF feature. For example, SVM classifier using topic feature outperforms SVM classifier using TF-IDF and improves the classification accuracy by up to 27%, which is consistent with the binary classification case. This again confirms that topic-based linguistic features are consistently more reliable and discriminative in multi-class user classification tasks. Second, in terms of accuracy, SVM classifier outperforms NB classifier by 23% using either TF-IDF feature or topic feature, which again shows that SVM performs better than NB in multi-class classification tasks.

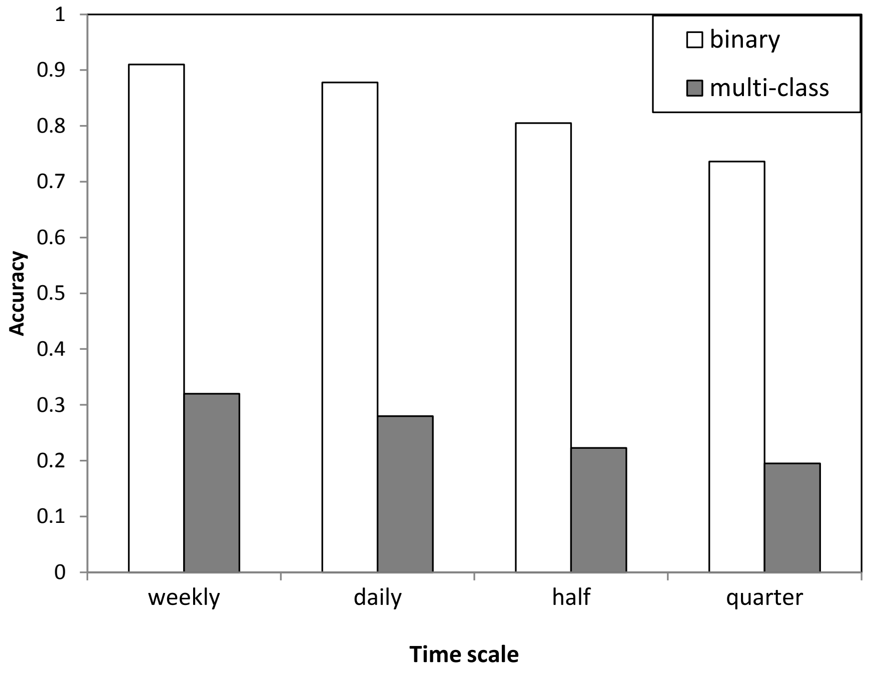
Figure 5.6 shows the performance of our proposed time-series based Algorithm 5 and Algorithm 6 for multi-class user classification. First, our 1NN classifier using DTW or SAX as distance functions show significant improvements over the basic methods in Figure 5.5. For example, our proposed All-At-Once classification algorithm using time series feature on a weekly basis and DTW as distance function outperforms SVM classifier using topic feature by improving the classification accuracy by around 39%. Second, our proposed One-Vs-All classification algorithm further improves the accuracy by 67% over the All-At-Once classification algorithm in the same setting and hence improves by 142% over the baseline. This is because our One-Vs-All classification algorithm builds K binary classifiers when classifying a new user, and returns the classifier producing the best result as the winner. Moreover, during the training of k^{th} binary classifier, the algorithm uses the category-specific keywords of k^{th} class to convert all the users in the training set into time series such that the inter-class difference among users from different categories can be amplified in order to boost the accuracy of classifying the new user into the positive class.

5.4.4 Impact of Temporal Feature Size

Next, we select the "best" binary and multi-class classifiers in time series domain as shown in Figures 5.3 and Figures 5.6, and further study the impact of temporal feature size in terms of classification accuracy. First, we choose different time periods ranging from 1, 2, 3, and 4 months. This represents different lengths of time series for classification. Second, in addition to daily and weekly time frames we used in converting tweet streams into time series, we further divide daily time frame into smaller segments, i.e., a half day and one quarter day. This represents different scales of time series generated for classification. Figure 5.7(a) compares the classification accuracy as a function of length of time series. We can clearly see that the performance of both binary and multi-class time series classifiers show similar patterns. First, as the length of time series increases, the accuracy of classification in time series domain increases accordingly. This is because the periodicity pattern in tweet streams tends to be steady in larger time periods. Second, the length of time series doesn't impact the accuracy of our time series classifiers too much even on shorter time periods, which demonstrates that our time series classifiers are robust. Figure 5.7(b) compares the classification accuracy as a function of time scale. First, as the time scale decreases, the accuracy of classification in time series domain decreases accordingly. This is because temporal variation in tweet streams can be aggregated in larger time scales, which in turn can amplify the inter-class difference. Second, using smaller time scale to convert into time series doesn't impact the accuracy of our time series classifiers too much, which again shows our time series classifiers are fairly reliable.



(a) Varying time length



(b) Varying time scale

Figure 5.7. Impact of temporal feature size for the best binary and multi-class classifier in time series domain.

5.5 Experiments on Political Interests

In order to corroborate our proposal, in this section, we further perform a classification task of user interests on a different data set. In this experiment, we aim at tackling the *binary* classification problem to identify users as either Democrats (i.e., left) or Republicans (i.e., right). Our experimental question is: How accurately can we predict if a Twitter user is a democrat or a republican?

We used the data set on political polarization from [91], which contains political communications during six-week period (Sep. 14 – Nov. 1, 2010) leading up to 2010 U.S. congressional midterm elections. A political communication is defined as any tweet containing at least one politically relevant *hashtag*. From the set of political tweets, two types of networks, i.e., mentions and retweets, are constructed among a set of Twitter users. Both networks represent public user interaction for political information flow. In this data set, each tweet has the timestamp and a set of hashtags available (no tweet messages available). Each user has the political affiliation information available (i.e., ground truth). Using only users with at least 30 retweet (RT) activities during the time period, at the end, our data set included 200 Democrats and 200 Republicans, a total of 14,952 retweets with 1,829 unique hashtags. The data set also provides 678 left-leaning (i.e., democrats) political hashtags (e.g., `#p2`, `#dadt`, `#healthcare`, `#hollywood`, `#judaism`, `#capitalism`, `#recession`, `#security`, `#dreamact`, `#publicoption`) and 611 right-leaning (i.e., republicans) political hashtags (e.g., `#tcot`, `#gop`, `#twisters`, `#israel`, `#foxnews`, `#sgp`, `#constitution`, `#patriots`, `#rednov`, `#abortion`). Note that we used only hashtags in this experiment.

Since the data set does *not* contain textual messages but only hashtags, the textual feature based baselines in Section 5.2 are not applicable. Instead, there-

Table 5.1. Democrats-specific (LEFT) and Republicans-specific (RIGHT) hashtags.

Category	Hashtag
LEFT	#p2, #dadt, #healthcare, #hollywood, #judaism, #capitalism, #recession, #security, #dreamact, #publicoption
RIGHT	#tcot, #gop, #twisters, #israel, #foxnews, #sgp, #constitution, #patriots, #rednov, #abortion

fore, we used two naive keyword-based (KB) classification as the baseline – i.e., the TAG+KB (resp. RT+KB) computes the percentage of hashtags (resp. retweets) containing category-related keywords and uses a simple threshold (e.g., 10%) to classify a user into the positive class. As to our proposed methods, we first used category-specific hashtags to convert each user's retweets into a time series on the daily time scale. We treat a user's daily retweet streams as a bag of hashtags and count the number of category-specific hashtags (DH) or category-specific retweets (DR) that appear in the daily retweets. We prepared two variations, using either democrats-specific (LEFT) and republicans-specific (RIGHT) hashtags to covert users into time series. Finally, we used the 1NN classifier with DTW as the distance function to do classification in time series domain. Same as Section 6.4, we measured the classification accuracy with 10-fold cross validation.

Figure 5.8 shows the comparison result. Similar to the results for sport interests in Section 6.4, our time series based classifiers outperform both heuristic baseline methods significantly. For instance, the best performing 1NN classifier using daily time series feature at hashtag level (LEFT+DH) increases the accuracy from the baselines using retweets (RT+KB) and hashtags (TAG+KB) by 16% and 22%, respectively. Next, regardless of using democrats-specific or republicans-specific hashtags, our time series classifiers at hashtag level (LEFT+DH or RIGHT+DH) outperforms the retweet-level classifiers (LEFT+DR or RIGHT+DR) . This is because there exist multiple category-specific hashtags in political retweets of a demo-

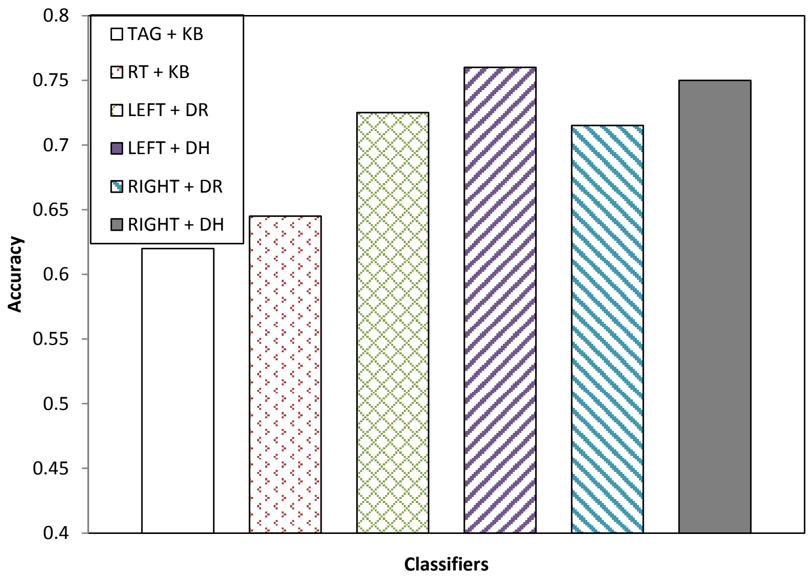


Figure 5.8. Binary classification results on political interests. Note that DR and DH are daily time series at retweet and hashtag levels, respectively.

crat or republican user such that the inter-class difference can be "amplified" when it is captured as time series based on the frequency of relevant political hashtags.

5.6 Summary

In this chapter, we presented a novel method to classify Twitter user interests using time series generated from the contents of tweet streams. By amplifying the latent *periodicity* pattern in tweets into time series, we showed the cases where both binary and multi-class classification accuracy can be improved significantly. Using real data sets on both sports and political interests, we validated our claim through comprehensive experiments by showing that our time series based classifiers outperform up to eight competing classification solutions significantly.

Chapter 6

Mining Social Activity Using Latent Topics

6.1 Overview

Online social networks (OSNs) have become popular platforms for news dissemination, professional networking, social recommendations, and online content curation. Millions of users connect to each other, express themselves and share interests through OSNs. For example, Facebook, LinkedIn and MySpace are social networks used to find and organize contacts while Flickr, YouTube and Instagram are used to share multimedia contents.

However, today's OSNs rely on users to manually post profiles consisting of attributes such as demographic information, geographic location and personal interests. This represents a significant burden on users who are members of multiple OSNs. As a result, not all users provide these attributes in their profiles which, in turn, reduces the usefulness of online social networking applications because such profile information is important for grouping users, sharing contents and recommending friends or products.

On the other hand, among the vast amount of user-generated contents, Facebook Likes activity is one of the highly available and public information in OSNs. Facebook Likes refer to the social activity by Facebook users to express their positive association with online contents such as photos, friends' status updates, products, sports, musicians, books, restaurants, or other popular websites [27]. Unlike other social activities, Facebook Likes are currently publicly available by default. Previous research has shown that 57% of Facebook user profiles publicly reveal at least one Like among different categories. This large amount of available activity information suggests that the majority of users consider this Facebook activity does not violate their privacy as there seems no correlation between their Likes and private data.

Thus, it is natural to try to utilize Facebook Likes provided by users in order to infer the missing user attributes in an online social network. Such ability of automatically predicting user attributes is very useful for many social networking applications such as friend recommendation and content sharing. Also it has less privacy concerns as users are more willing to publicly reveal their Likes activity in online social networks.

In this chapter, we study the problem of predicting user attributes to pre-defined categories based on the activity of their Facebook Likes. Formally, we study the following:

Definition (User Attribute Prediction) *Given a set of Facebook users U, a set of Facebook Like items I, a $|U| \times |I|$ Likes matrix $\mathbf{L} = [\mathbf{l}_1, ..., \mathbf{l}_{|U|}]^T$, where $l_{u,i} \in \{0, 1\}$ represents user $u \in U$ likes item $i \in I$ or not, a pre-defined set of K attribute labels $A = \{a_1, ..., a_K\}$, and labeled samples such that $\langle u, a\rangle \in U \times A$,* *predict and assign an attribute label to a unlabeled user v with $|I|$-dimensional Likes vector* $\mathbf{l}_v$. □

Some researchers have already studied the problem of predicting private traits and attributes from digital records of human behavior. Recent study [27] has shown that Facebook Likes can be used to automatically and accurately predict sensitive personal attributes, such as sexual orientation, ethnicity, religious and political views, intelligence, happiness, drug use, parental separation, age, and gender. Based on demographic profiles and results of psychometric tests as well as their Facebook Likes data from 58,466 volunteers, their study used regression models to predict individual psychodemographic profiles from Facebook activities. The method can achieve best discriminative results for predicting dichotomous variables such as gender and ethnicity. And the authors claimed that even for numerical variables such as openness attribute from personality traits, the prediction accuracy is also close to the accuracy of a standard personality test.

In this chapter, we introduce our topic modeling approach to tackle the problem of user attribute prediction. In particular, we propose a LDA framework to extract topic features from Facebook Likes activity such that the prediction problem can be solved using latent topics. The efficacy of our proposed approach is validated through comprehensive experiments.

6.2 Predicting User Attributes using Activity Features

We first present the baseline models for predicting user attributes based on the *activity features* extracted from Facebook Likes. Given a set of Facebook activities, we represent each user as a vector of likes and directly extract activity features from the users.

6.2.1 Feature Selection

We use Principal Component Analysis (PCA) to select reduced feature vectors based on Facebook Likes: Principal Components of Likes from Singular-Value Decomposition (SVD).

PCA by SVD. The Principal Component Analysis [92] is an unsupervised dimension reduction technique and seeks a projection that can best represent the data in the reduced space. Assume $\mathbf{x}$ (with components x_j, $j = 1, ..., n$) is a Facebook Likes feature vector with probability distribution $P(x)$. Let $\{\mathbf{x}_\alpha | \alpha = 1, ...m\}$ be a sample from $P(x)$, which form the $n \times m$ data matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m]$.

PCA is based on the first and the second empirical moments of the sample data matrix. The mean vector,

$$\langle \mathbf{x} \rangle = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i$$

and the empirical covariance matrix,

$$\mathbf{C} = \frac{1}{m} \sum_{i=1}^{m} (\mathbf{x}_i - \langle \mathbf{x} \rangle)(\mathbf{x}_i - \langle \mathbf{x} \rangle)^T$$

Using the matric formulation, we have

$$\mathbf{C} = \frac{1}{m} \mathbf{X}\mathbf{X}^T$$

where the mean of the data has been removed so that the matrix will not be affected by the location of the center of the data.

PCA is to find the directions in the data with the most variation, i.e., the eigenvectors corresponding to the largest eigenvalues of the covariance matrix, and to project the data onto these directions. Let's denote the matrix of eigenvectors sorted according to eigenvalue by $\widetilde{\mathbf{U}}$, then the PCA transformation of the data is

$$\mathbf{Y} = \widetilde{\mathbf{U}}^T \mathbf{X}$$

The eigenvectors are called the principal components. By selecting only the first d rows of $\mathbf{Y}$, we can project the original data from n dimensions to d dimensions.

We can use Singular-Value Decomposition (SVD) to perform PCA. By decomposition using SVD, we have

$$\mathbf{X} = \mathbf{U}\Gamma\mathbf{V}^T$$

and the covariance matrix can be written as

$$\mathbf{C} = \frac{1}{m}\mathbf{X}\mathbf{X}^T = \frac{1}{m}\mathbf{U}\Gamma^2\mathbf{U}^T$$

where $\mathbf{U}$ is a $n \times m$ matrix with orthonormal columns ($\mathbf{U}^T\mathbf{U} = \mathbf{I}$), while $\mathbf{V}$ is a $m \times m$ matrix with orthonormal columns ($\mathbf{V}^T\mathbf{V} = \mathbf{I}$), and Γ is a $m \times m$ diagonal matrix with positive or zero elements called the singular values.

The transformed data in terms of the SVD decomposition of $\mathbf{X}$ can thus be written as

$$\mathbf{Y} = \widetilde{\mathbf{U}}^T \mathbf{X} = \widetilde{\mathbf{U}}^T \mathbf{U}\Gamma\mathbf{V}^T$$

where $\widetilde{\mathbf{U}}^T\mathbf{U}$ is a simple $n \times m$ matrix which is one on the diagonal and zero elsewhere.

6.2.2 Prediction Models

We select three popular models over the activity features: Naive Bayes (NB), Support Vector Machines (SVM) and Logistic Regression (LR).

Naive Bayes. In this chapter, we adopt a multinomial Naive Bayes model. Given the user attribute prediction problem having K labels $\{a_1,\ a_2\ ...,a_K\}$ with prior probabilities $P(a_1)$,...,$P(a_K)$, we assign an attribute label a to a Facebook user u with Likes feature vector $\mathbf{l} = (l_1, l_2..., l_N)$, such that

$$a = \arg\max_{a} P(a = a_k | l_1, l_2..., l_N)$$

That is to assign the attribute label with the maximum a posterior probability given the observed data. This posterior probability can be formulated using Bayes theorem as follows:

$$P(a = a_k | l_1, l_2..., l_N) = \frac{P(a_k) \times \prod_{i=1}^{N} P(l_i|a_k)}{P(l_1, l_2..., l_N)}$$

where the objective is to assign a given user u having a feature vector $\mathbf{l}$ consisting of N features to the most probable attribute label. $P(l_i|a_k)$ denotes the conditional probability of feature l_i found in Facebook Likes of user u given the attribute label a_k.

Support Vector Machines. Given a corpus of U Facebook users and attribute labels for training $\{(\mathbf{l}_u, a_u)|u = 1, ..., U\}$, where $\mathbf{l}_u$ is the Likes feature vector of user u and a_u is the target attribute label, SVM maps these input feature vectors into a high dimensional reproducing kernel Hilbert space, where a linear machine is constructed by minimizing a regularized functional. The linear machine takes the form of $\varphi(\mathbf{l}) = \langle \mathbf{w} \cdot \phi(\mathbf{l}) \rangle + b$ where $\phi(\cdot)$ is the mapping function, b is the bias and the dot product $\langle \phi(\mathbf{l}) \cdot \phi(\mathbf{l}') \rangle$ is also the kernel $K(\mathbf{l}, \mathbf{l}')$. The regularized functional is defined as:

$$R(\mathbf{w}, b) = C \cdot \sum_{u=1}^{U} \ell(a_u, \varphi(\mathbf{l}_u)) + \frac{1}{2}\|\mathbf{w}\|^2$$

where the regularization parameter $C > 0$, the norm of $\mathbf{w}$ is the stabilizer and $\sum_{u=1}^{U} \ell(a_u, \varphi(\mathbf{l}_u))$ is empirical loss term. In standard SVM, the regularized functional can be minimized by solving a convex quadratic optimization problem which guarantees a unique global minimum solution.

Logistic Regression. The Logistic Regression, like the Naive Bayes, is a very popular and widely used classification technique. While Naive Bayes is a generative model to form a statistical model for each class, Logistic Regression is a discriminative model. Given the user attribute prediction problem having K labels $\{a_1, a_2 ..., a_K\}$, a Facebook user u with Likes feature vector $l = (l_1, l_2..., l_N)$, and a parameter vector θ, for binary classification with -1, 1 class coding, we define the label probability via the logistic function:

$$P(a|l) = \frac{1}{1 + \exp(-a\theta^{\mathsf{T}} l)}$$

Logistic regression can be easily generalized to K multiple classes with each class has its own parameter θ_k, we then define the label probability via the softmax function:

$$P(a = a_k|l) = \frac{\exp(\theta_k^{\mathsf{T}} l)}{\sum_{i=1}^{K} \exp(\theta_i^{\mathsf{T}} l)}$$

Finding the parameter θ can be done by maximizing the conditional log likelihood of the training data $(l, a)_{1:n}$:

$$\arg\max_{\theta} \sum_{i=1}^{n} \log p(a_i | l_i, \theta)$$

6.3 Predicting User Attributes using Latent Topics

In this section, we introduce our LDA-based approach to tackle the problem of user attribution prediction. In the following, we give the general problem formulation for Facebook Likes Network (LN) and topic-based generative modeling in terms of LN.

6.3.1 Problem Formulation

We define the following terms in this chapter:

- A user u is a sequence of N Facebook Like items denoted by $\mathbf{l} = \{l_1, l_2, ..., l_N\}$, where l_n denotes the nth like item of the user.
- A corpus is a group of M users denoted by $U = \{u_1, u_2, ..., u_M\}$.
- A vocabulary of Likes is a set of unique like items in a corpus denoted by $L = \{l_1, l_2, ..., l_p\}$ with size p.
- The relationships between users and likes are connected by a set of latent variables $Z = \{z_1, z_2, ..., z_T\}$ with size T, each of which represents a latent topic.

The Likes Network (LN) is formally defined as:

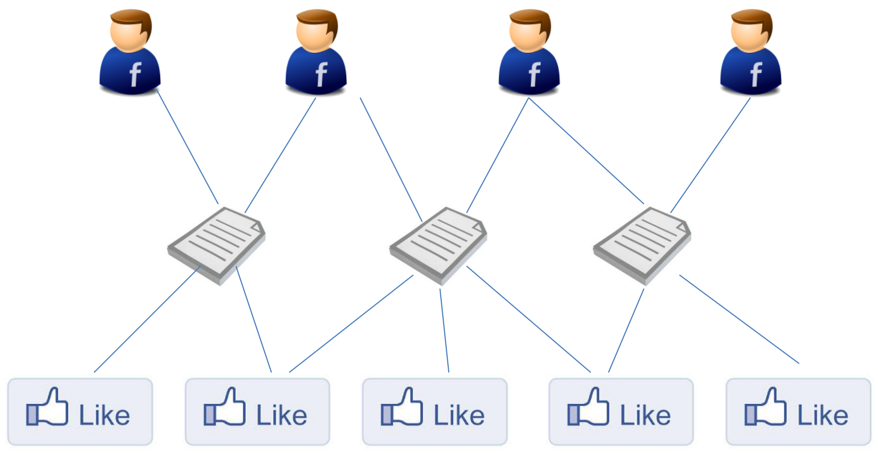


Figure 6.1. A connectivity graph of users, topics and likes in Likes Network.

Definition (Likes Network) *A Likes Network is defined as a bipartite graph $G = (V, E)$ where V is the set of vertices V and it contains two classes X and Y such as $V = X \cup Y$ and $X \cap Y = \emptyset$, each edge $e_{i,j} \in E$ has one endpoint (i) in X and the other endpoint (j) in Y. X represents a set of users while Y represents a set of likes items. An edge $(x, y) \in E$ from user x to item y indicates the user x "likes" the item y.* $\square$

We assume that there is a set of latent topics existing in the Likes Network such that a set of likes are most likely co-occurred in a specific topic, and we define a topic as:

Definition (Topic) *A semantically coherent topic ϕ in a Likes Network G is defined as a multinomial distribution of all like items in L, i.e., $\{p(l|\phi)\}_{l \in L}$ with the constraint $\sum_{l \in L} p(l|\phi) = 1$.* $\square$

Based on the definitions of these concepts, we give the problem formulation as follows:

Definition (Topic-Likes Modeling) *Given a Likes Network G, model and extract a set of T topics $\{\phi_1, ..., \phi_T\}$ where ϕ_i is a topic in G.* $\square$

In our user-topic-likes scenario, an observation is treated as a tuple $\{u, l\}$ that represents an instance that a user u likes an item l. The relationship inherent in the tuples is associated by a set of topics Z. Our mixture model has a conditional independence assumption of variables, i.e., the observed variables are conditionally independent on the state of the underlying latent variable, which are essentially related to users' attributes. Specifically, a user u is a mixture of several topics in Z with different probabilities, and the latent variables consequently generate a set of likes $\mathbf{l}$ that are most likely co-occurred in a specific topic. Figure 6.1 shows the graphical illustration of users, topics and likes.

6.3.2 User-level LDA model

In this section, we introduce our user-level LDA model for activity-based user attribute prediction in a Likes Network. While LDA represents documents as bags of words, we represent users as their Facebook Likes. Therefore, a Facebook user is represented as a multinomial distribution over hidden topics. Given a number U of Facebook users and a number T of topics, each user u is represented by a multinomial distribution θ_u over topics, which is drawn from a Dirichlet prior with parameter α. A topic is represented by a multinomial distribution ϕ_t over likes, which is drawn from another Dirichlet prior with parameter β.

The notations are shown in Table 6.1. θ_u denotes a T-dimensional probability vector and represents the topic distribution of the user u. ϕ_t denotes a L-dimensional probability vector where $\phi_{t,l}$ specifies the probability of generating Facebook like l given topic t. $Multi(.)$ denotes multinomial distribution. $Dir(.)$ denotes Dirichlet distribution. α is a T-dimensional parameter vector of the Dirichlet distribution over θ_u, and β is a L-dimensional parameter vector of the Dirichlet

Table 6.1. Notations

Symbol	Description
U	total number of Facebook users
L	total number of Facebook like items
T	total number of topics
N_u	total number of like items of user u
$l_{u,i}$	ith like item of user u
$z_{u,i}$	latent topic at ith like item in Facebook likes of user u
$\theta_{u,i}$	probability of topic i of user u
$\phi_{t,l}$	probability of like item l in topic t

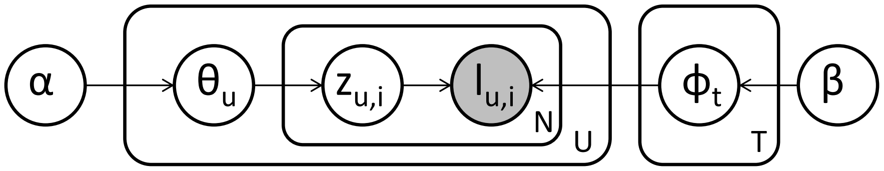


Figure 6.2. User-level LDA Model.

distribution over ϕ_t. The generative process is shown in Algorithm 7.

The topic vector acts as a low-dimensional feature representation of users' Facebook Likes and can be used as input into any prediction algorithm. In other words, for each user u, we can use our proposed user-level LDA model to learn θ_u for that user and then treat θ as the features in order to do user attribute prediction. We still apply Naive Bayes, Support Vector Machines and Logistic Regression models in the reduced dimensional space.

Algorithm 7: **User-level LDA model for Facebook Likes**.

```
1 For each of the $T$ topics $t$, sample a multinomial distribution $\phi_t$ from a Dirichlet
  distribution with prior $\beta$;
2 For each of the $U$ users $u$, sample a multinomial distribution $\theta_u$ from a Dirichlet
  distribution with prior $\alpha$;
3 For each like item $l_{u,i}$ of user $u$, sample a topic $z_{u,i}$ from a multinomial
  distribution with parameter $\theta_u$;
4 Sample like $l_{u,i}$ from a multinomial distribution with parameter $\phi_{z_{u,i}}$.
```

6.4 Experiments on Demographical Attributes

In order to validate our proposed approach, we apply it to both binary and multi-class user attribute prediction with respect to identifying Facebook users' personal demographical information. Specifically, our experimental questions are the following:

1. *Binary*: How accurately can we predict a Facebook user's gender and political views respectively?
2. *Multi-class*: How accurately can we predict a Facebook user's age group and relationship status respectively?

6.4.1 Set-Up

We used data on US Facebook users' psychodemographic profiles and their lists of Likes from [27]. The original data were obtained from the myPersonality application[1]. Volunteer users provided their data for this study and gave their consent to have their scores and profile information recorded for analysis. We selected 1609 Facebook users who declared their gender, age, relationship status and political view in their profiles (i.e., ground truth). In particular, gender (i.e.,"male" or "female") and political view (i.e., "Democrat" or "Republican") are two binary attributes. Figure 6.3(a) and Figure 6.3(b) show the distribution of gender and political view attributes respectively. Relationship status attribute recorded from the user profile has ten classes, where the options are "Single", "In a Relationship", "Married", "Engaged", "Complicated", "Open", "Widowed", "Divorced", "Separated" and "Partnership". Age attribute was originally recorded as a numerical

[1] http://www.mypersonality.org

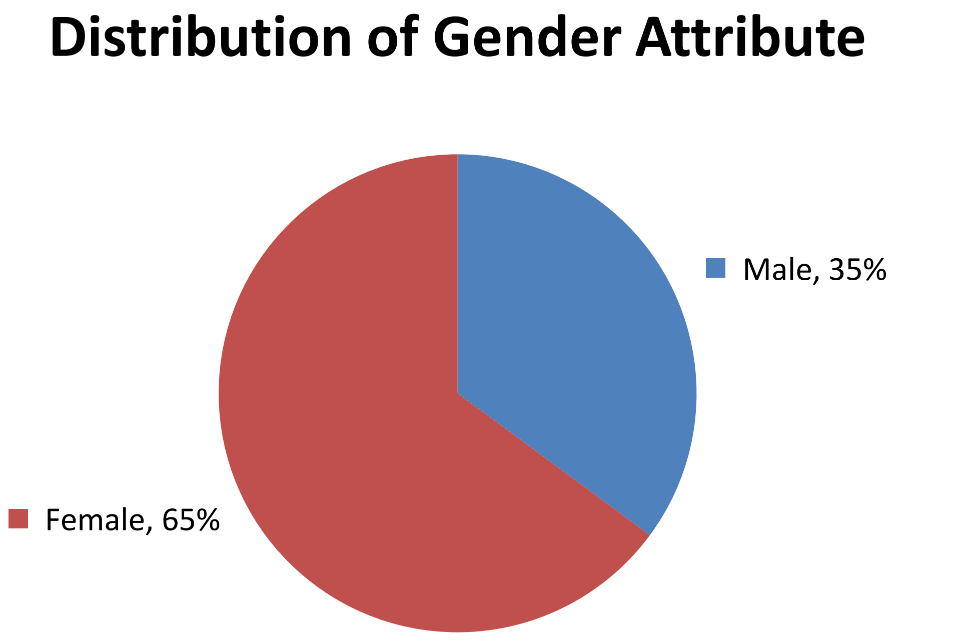
variable, and we further divided the users into five age groups, where the groups are “less than 20 years old”, “between 20 and 30 years old”, “between 30 and 40 years old”, “between 40 and 50 years old” and “more than 50 years old”. Figure 6.4(a) and Figure 6.4(b) show the distribution of age and relationship status attributes respectively.

Evaluation Metrics. For evaluation purpose, all the users can be grouped into four categories, i.e., true positive (TP), true negatives (TN), false positives (FP) and false negatives (FN). For example, the true positives are the users that belong to positive class and are in fact predicted to the positive class, and the false positives are the users not belonging to positive class but incorrectly predicted to the positive class. Since we are interested in both positive and negative classes especially in multi-class prediction, we use the *accuracy (ACC)* metric to measure the performance as follows:

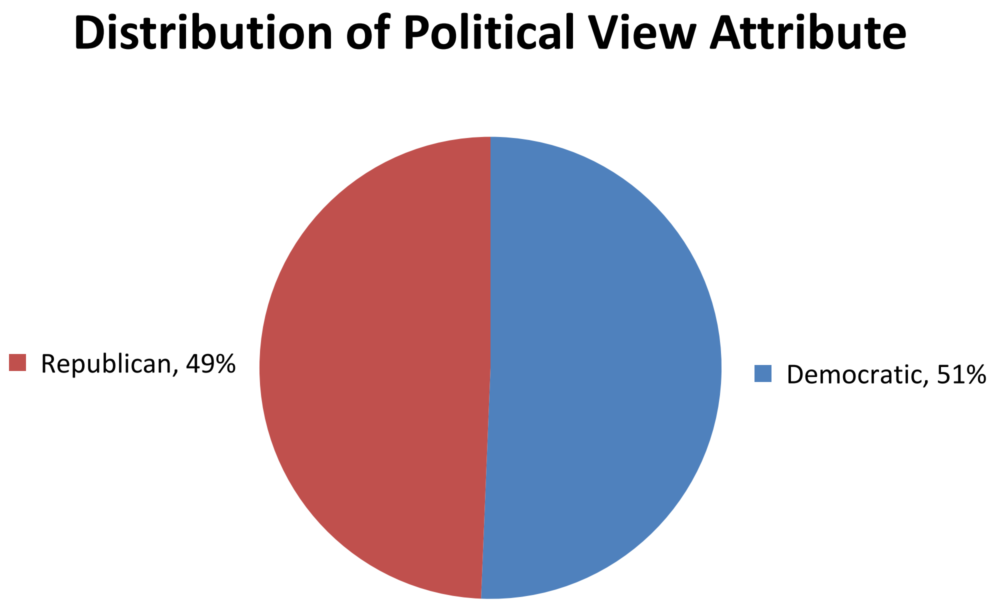
$$ACC = \frac{TP + TN}{TP + FP + TN + FN}$$

In all subsequent experiments, we use the *10-fold cross validation* [25] to measure the accuracy.

Baseline Method. We use SVD in Section 6.2 to reduce the dimensionality of the user-likes matrix in order to facilitate the predictive analysis, i.e., each user is represented as a vector of principle component scores. As the baseline, SVD provides a low-rank approximation to the original matrix. We use the NB, SVM and LR based prediction models using the activity features in Section 6.2 as the baseline. In the following, we compare the accuracy of our proposed topic-based prediction method against the baseline.

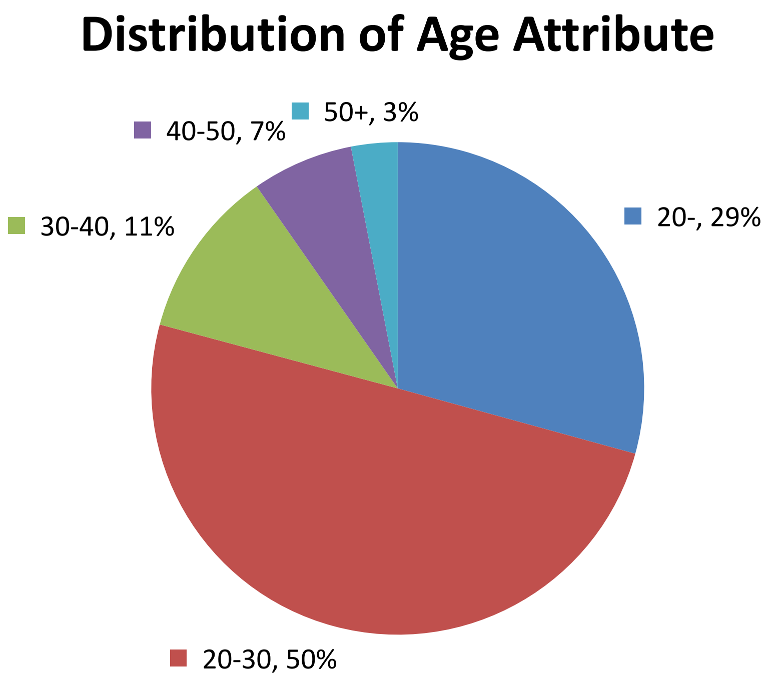


(a) Gender

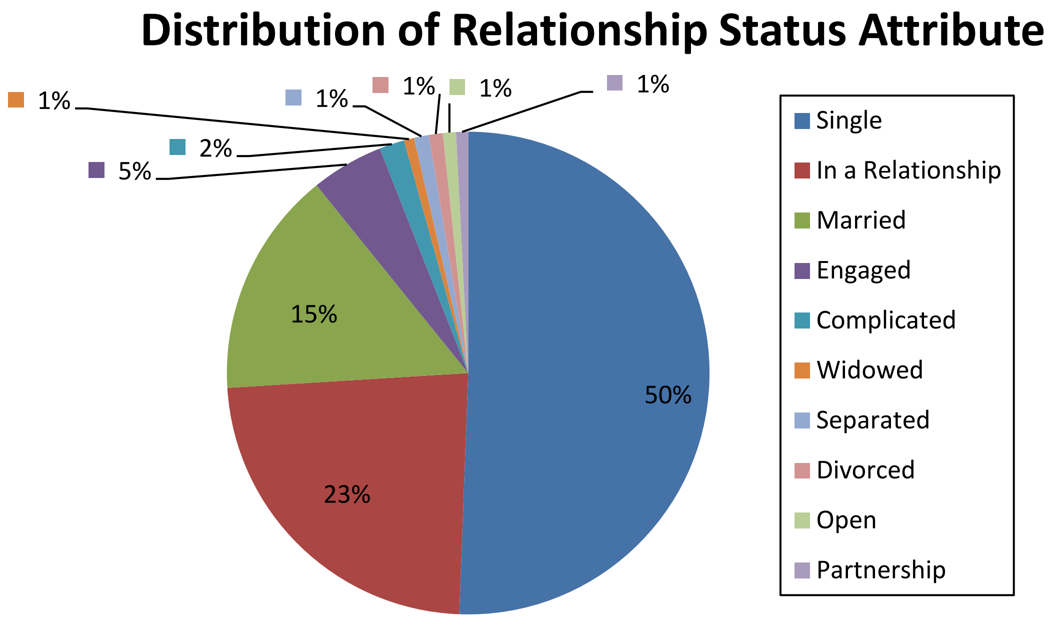


(b) Political View

Figure 6.3. Percentages of Facebook users on two binary attributes.



(a) Age



(b) Relationship Status

Figure 6.4. Percentages of Facebook users on two multi-class attributes.

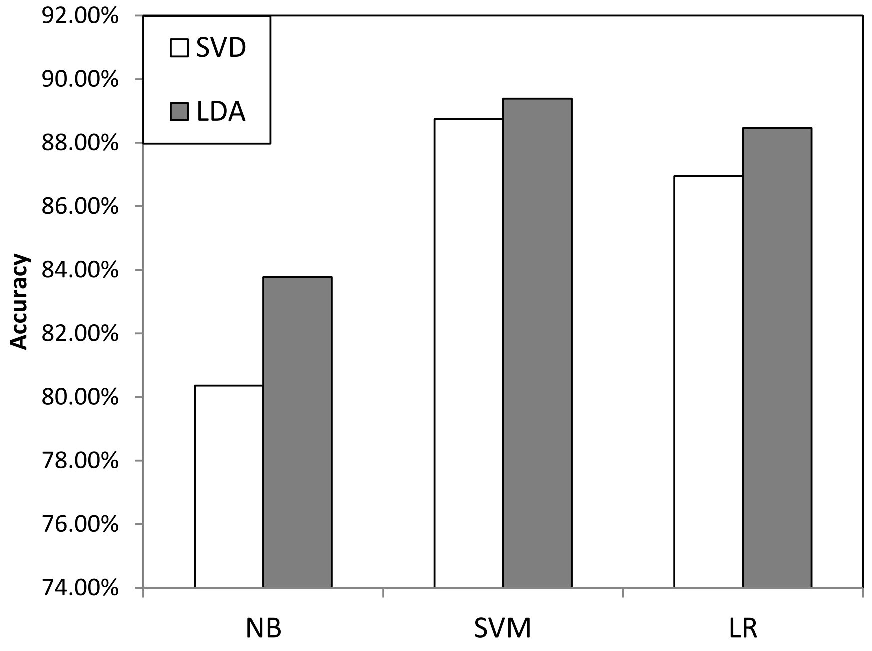


Figure 6.5. Gender prediction of Facebook users using Likes.

6.4.2 Binary Attribute Prediction

Given a set of Facebook Likes from a user, the goal of our binary attribute prediction is to predict: (a) *gender* of the user, i.e., whether the user is a female or male; (b) *political view* of the user, i.e., whether the user is a democrat or republican.

We used three different baseline approaches SVD+NB (resp. SVD+SVM and SVD+LR), all of which use SVD with 100 components to obtain activity features after reducing the dimensionality of the user-likes matrix. As to our proposed methods, we first used the user-level LDA model with 100 topics to extract topic features from Facebook Likes activity. Then we applied three variations LDA+NB (resp. LDA+SVM and LDA+LR) to predict the user attributes.

Gender. Figure 6.5 shows the prediction accuracy of baselines and our methods for gender attribute. First, we can clearly see that all methods achieve fairly high accuracy (e.g., above 80%). This demonstrates the high predictive power of Likes

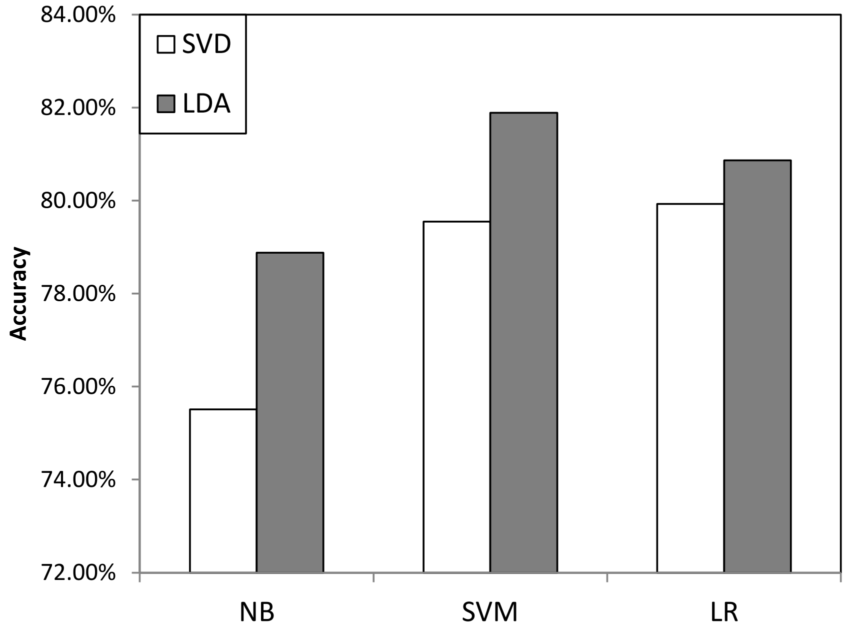


Figure 6.6. Political View prediction of Facebook users using Likes.

activity to discriminate the gender of Facebook users. Second, we can see that for each of the three prediction models, the topic features derived from our topic modeling approach outperform the activity features using SVD, which shows topic-based linguistic features are consistently reliable and more discriminative in general classification and prediction tasks. Third, using either SVD component features or topic features, SVM model generally outperforms NB model and LR model, which shows that SVM performs better than NB and LR in general classification and prediction tasks [86].

Political View. Figure 6.6 shows the prediction accuracy of baselines and our methods for political view attribute. First, we can see that although both attributes (i.e., gender and political view) are binary, all prediction methods achieve not as high accuracy as in the previous gender prediction. This indicates less predictive power of Likes activity to discriminate the political view of Facebook users

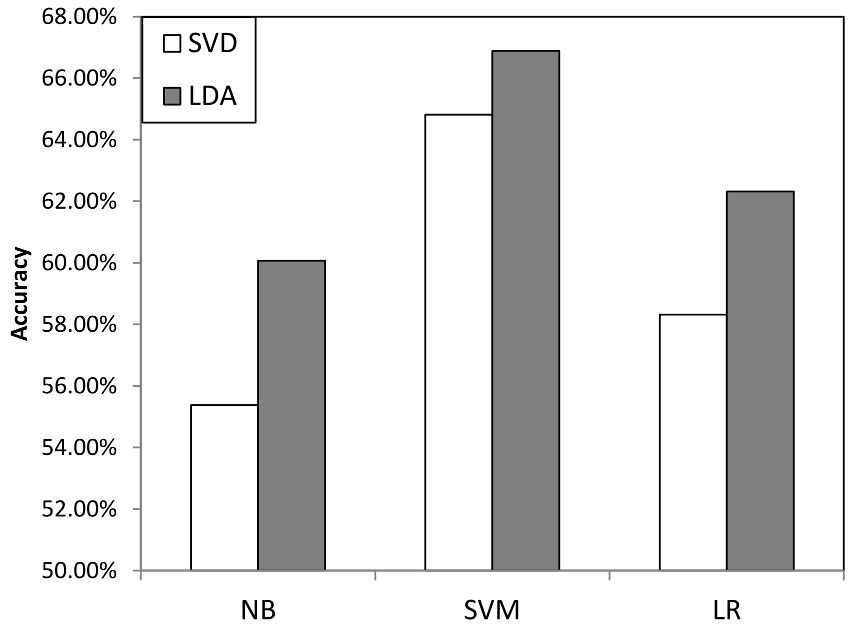


Figure 6.7. Age prediction of Facebook users using Likes.

as compared to the gender attribute. Second, we can see that similar to the gender case, the topic features derived from our topic modeling approach outperform the activity features using SVD, which again shows topic-based linguistic features are consistently reliable and more discriminative. Third, using either SVD component features or topic features, SVM model generally outperforms NB model and LR model, which shows similar pattern in terms of prediction performance.

6.4.3 Multi-class Attribute Prediction

Next, the goal of multi-class prediction is to predict: (a) *age* of the user, i.e., which age group (out of 5 groups) the user is belonging to ; (b) *relationship status* of the user, i.e., which relationship (out of 10 relationships) the user is currently in.

Similar to Section 6.4.2, we applied 3 baselines using SVD component features and 3 variations of our proposal using topic features.

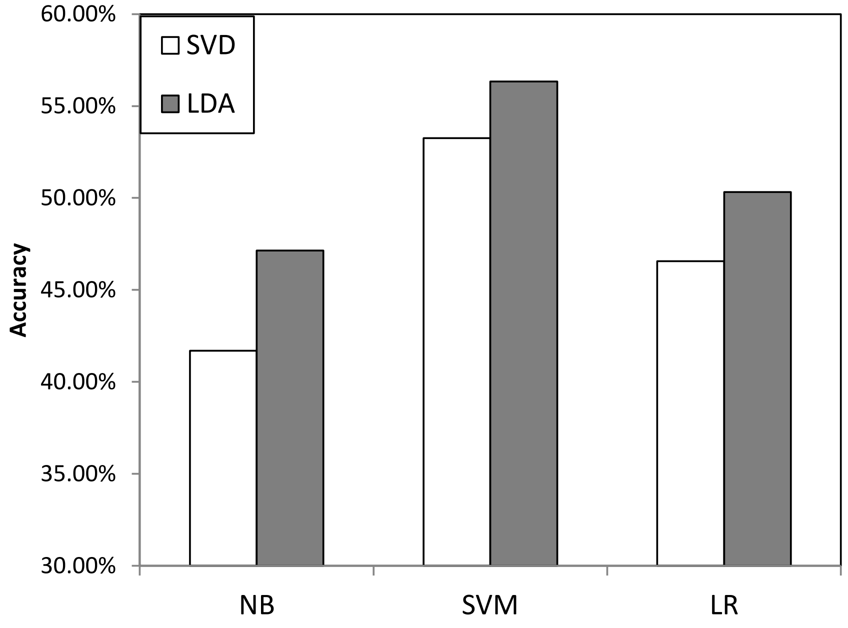


Figure 6.8. Relationship Status prediction of Facebook users using Likes.

Age. Figure 6.7 compares the multi-class prediction accuracy among baseline methods and our proposed methods for age attribute. First, regarding different features extracted from Facebook Likes, it is shown that all models using the topic features derived from our topic modeling approach outperform the models using the SVD component features. For example, NB model using topic features outperforms NB model using activity features and improves the prediction accuracy by 9%, which is consistent with the binary prediction case. This again confirms that topic-based linguistic features are consistently more reliable and discriminative in *multi-class* user classification and prediction tasks. Second, in terms of accuracy, SVM model outperforms NB or LR model by up to 17% using either activity features or topic features, which again shows that SVM performs better than NB and LR in *multi-class* classification and prediction tasks.

Relationship Status. Figure 6.8 compares the multi-class prediction accuracy among baseline methods and our proposed methods for relationship status attribute. First, we can clearly see that all methods achieve fairly low accuracy (e.g., below 50%). This indicates the less predictive power of Likes activity to discriminate the relationship status of Facebook users as compared to the age attribute. Second, regarding different features extracted from Facebook Likes, it is again shown that all models using the topic features derived from our topic modeling approach outperform the models using the SVD component features. For example, NB model using topic features outperforms NB model using activity features and improves the prediction accuracy by 13%, which is consistent with the binary prediction case. Third, in terms of accuracy, SVM model outperforms NB or LR model by up to 28% using either activity features or topic features, which shows similar pattern in terms of prediction performance.

6.5 Summary

In this chapter, we presented a user-level LDA model approach to predicting Facebook user attributes using topic features extracted from Facebook Likes activity. By semantically modeling the relationship between users and their social activity, we have shown the cases where both binary and multi-class prediction accuracy can be improved effectively. Using real data sets on four demographical attributes of Facebook users, we validated our claim through comprehensive experiments by showing that our prediction models using latent topics outperform three competing prediction solutions.

Chapter 7

Conclusion and Future Work

7.1 Contributions

In this dissertation, we have presented a series of new methods for mining texts and social users using time series and latent topics. We applied our methods to four data mining applications, e.g., record linkage, document modeling, mining social content and mining social activity, and showed the performance of our knowledge discovery solutions.

On mining texts using time series in terms of record linkage, our contributions include: (1) As a solution to information-sensitive text mining in time series domain, we formally propose our T^3 framework to map string/text to time series and show how to apply it to the record linkage problem. (2) At character level, we propose and evaluate nine different combinations of n-grams and space-filling curve techniques to translate string data into time series data. At word level, we propose and evaluate three n-grams during numeric assignments while considering the relative importance of tokens in the string data. (3) We apply the T^3 framework to the record linkage problem and compare the performance against popular approximate pattern matching schemes such as Levenshtein and report promising results of our proposal in terms of both speed and accuracy.

On mining texts using latent topics in terms of document modeling, our contributions include: (1) As a solution to tackling noisy text data problems in document modeling, we formally incorporate textual errors into the document generation process and show how to apply it to the model formulation. (2) We discard the bag-of-words assumption in the LDA model. Instead, we assume that successive words in the document are more likely to have the same topic. We model the topics in a document to form a Markov chain with a transition probability and show how to incorporate dependency of topics into the generative process. (3) We apply our proposed models to both real and synthetic data sets and compare the performance against the traditional LDA model and the state-of-the-art N-Grams model, and report promising results of our proposal in terms of perplexity.

On mining social content using time series in terms of social user mining, our contributions include: (1) As a solution to the Twitter user classification problem in time series domain, we formally propose our framework to map users to time series for classification. (2) We formulate the problem of user classification as a document categorization problem in the Twitter setting, and show the procedure of feature selection as well as the detailed evaluation of different classifiers. (3) We validate our approach in in both binary and multi-class Twitter user classification settings and successfully demonstrate that our proposal substantially outperforms eight competing methods in identifying Twitter users with certain sports and political interests.

On mining social activity using latent topics in terms of social user mining, our contributions include: (1) As a solution to predicting social user attributes using latent topics, we formally propose our topic modeling framework based on

Facebook activities. (2) We formulate the problem of predicting user attributes using Facebook Likes, and show the procedure of feature selection and dimension reduction as well as the detailed evaluation of different prediction models. (3) We validate our topic modeling approach in both binary and multi-class Facebook user attribution prediction and show that our proposal can effectively improve the performance of baseline methods in predicting four demographical attributes of Facebook users.

7.2 Future Work

Many directions are ahead for future research.

On record linkage, first we plan to extend our T^3 framework to other text mining areas such as document clustering and classification. The sizes and dimensions of the data increase dramatically when documents are considered. Second, more sophisticated transformation schemes and advanced similarity functions need to be devised to provide comparable accuracy using time series data to their counterpart using text data.

On document modeling, first we plan to infer more complex topic structures and conduct tests of statistically significant differences across all the models. Second, we plan to apply our proposed models to handling textual errors in user-generated contents on social media.

On social user mining, first we will explore other feature selection techniques in time series domain and investigate more sophisticated transformation schemes to incorporate content features. Second, we plan to integrate other activity features and incorporate user attribute as an additional latent variable in supervised generative modeling in order to further improve classification and prediction accuracy.

Bibliography

[1] Walker, D. D., W. B. Lund, and E. K. Ringger (2010) "Evaluating Models of Latent Document Semantics in the Presence of OCR Errors," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 240–250.

[2] Java, A., X. Song, T. Finin, and B. L. Tseng (2007) "Why We Twitter: An Analysis of a Microblogging Community," in *Proceedings of the 9th International Workshop on Knowledge Discovery on the Web*, pp. 118–138.

[3] Wick, M. L., M. G. Ross, and E. G. Learned-Miller (2007) "Context-Sensitive Error Correction: Using Topic Models to Improve OCR," in *Proceedings of the 9th International Conference on Document Analysis and Recognition*, pp. 1168–1172.

[4] Tan, P.-N., M. Steinbach, and V. Kumar (2006) *Introduction to Data Mining*, Pearson Addison Wesley.

[5] Han, J., M. Kamber, and J. Pei (2011) *Data Mining: Concepts and Techniques*, Morgan Kaufmann.

[6] Wu, X., V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. F. M. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg (2008) "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, **14**(1), pp. 1–37.

[7] Witten, I. H., E. Frank, and M. A. Hall (2011) *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann.

[8] Zhao, Z. A. and H. Liu (2012) *Spectral Feature Selection for Data Mining*, Chapman & Hall.

[9] Liu, H. and H. Motoda (2008) *Computational Methods of Feature Selection*, Chapman & Hall.

[10] Mislove, A., B. Viswanath, P. K. Gummadi, and P. Druschel (2010) "You are who you know: inferring user profiles in online social networks," in *Proceedings of the 3rd International Conference on Web Search and Web Data Mining*, pp. 251–260.

[11] Winkler, W. E. (1999) *The State of Record Linkage and Current Research Problems*, *Tech. rep.*, US Bureau of the Census.

[12] Faloutsos, C., M. Ranganathan, and Y. Manolopoulos (1994) "Fast Subsequence Matching in Time-Series Databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 419–429.

[13] Blei, D. M., A. Y. Ng, and M. I. Jordan (2003) "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, **3**, pp. 993–1022.

[14] Rosen-Zvi, M., T. L. Griffiths, M. Steyvers, and P. Smyth (2004) "The Author-topic Model for Authors and Documents," in *Proceedings of the 20th Conference in Uncertainty in Artificial Intelligence*, pp. 487–494.

[15] Chen, X., C. Lu, Y. An, and P. Achananuparp (2009) "Probabilistic Models for Topic Learning from Images and Captions in Online Biomedical Literatures," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pp. 495–504.

[16] Liu, Y., A. Niculescu-Mizil, and W. Gryc (2009) "Topic-link LDA: Joint Models of Topic and Author Community," in *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 665–672.

[17] Newman, D., C. Chemudugunta, and P. Smyth (2006) "Statistical Entity-topic Models," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 680–686.

[18] Steyvers, M., P. Smyth, M. Rosen-Zvi, and T. L. Griffiths (2004) "Probabilistic Author-topic Models for Information Discovery," in *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 306–315.

[19] Blei, D. M. and J. D. Lafferty (2006) "Dynamic Topic Models," in *Proceedings of the 23rd International Conference on Machine Learning*, pp. 113–120.

[20] Nallapati, R., A. Ahmed, E. P. Xing, and W. W. Cohen (2008) "Joint Latent Topic Models for Text and Citations," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 542–550.

[21] Lund, W. B. and E. K. Ringger (2009) "Improving optical character recognition through efficient multiple system alignment," in *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pp. 231–240.

[22] Chen, Y., D. Pavlov, and J. F. Canny (2009) "Large-scale behavioral targeting," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 209–218.

[23] Rao, D., D. Yarowsky, A. Shreevats, and M. Gupta (2010) "Classifying Latent User Attributes in Twitter," in *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, pp. 37–44.

[24] Pennacchiotti, M. and A.-M. Popescu (2011) "A Machine Learning Approach to Twitter User Classification," in *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*.

[25] ——— (2011) "Democrats, Republicans and Starbucks Afficionados: User Classification in Twitter," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 430–438.

[26] Yang, J. and J. Leskovec (2011) "Patterns of temporal variation in online media," in *Proceedings of the 4th International Conference on Web Search and Web Data Mining*, pp. 177–186.

[27] Kosinski, M., D. Stillwell, and T. Graepel (2013) "Private traits and attributes are predictable from digital records of human behavior," in *Proceedings of the National Academy of Sciences*.

[28] Viswanath, B., A. Mislove, M. Cha, and P. K. Gummadi (2009) "On the Evolution of User Interaction in Facebook," in *Proceedings of the 2nd ACM Workshop on Online Social Networks*, pp. 37–42.

[29] Bilenko, M., R. J. Mooney, W. W. Cohen, P. D. Ravikumar, and S. E. Fienberg (2003) "Adaptive Name-Matching in Information Integration," *IEEE Intelligent Systems*, **18**(5), pp. 16–23.

[30] Fellegi, I. P. and A. B. Sunter (1969) "A Theory for Record Linkage," *Journal of the American Statistical Association*, **64**(328), pp. 1183–1210.

[31] Hernández, M. A. and S. J. Stolfo (1995) "The Merge/Purge Problem for Large Databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 127–138.

[32] On, B.-W., D. Lee, J. Kang, and P. Mitra (2005) "Comparative Study of Name Disambiguation Problem using a Scalable Blocking-based Framework," in *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pp. 344–353.

[33] Chaudhuri, S., K. Ganjam, V. Ganti, and R. Motwani (2003) "Robust and Efficient Fuzzy Match for Online Data Cleaning," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 313–324.

[34] Sarawagi, S. and A. Bhamidipaty (2002) "Interactive Deduplication using Active Learning," in *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 269–278.

[35] Warner, J. W. and E. W. Brown (2001) "Automated Name Authority Control," in *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pp. 21–22.

[36] Gravano, L., P. G. Ipeirotis, N. Koudas, and D. Srivastava (2003) "Text Joins in an RDBMS for Web Data Integration," in *Proceedings of the 12th International World Wide Web Conference*, pp. 90–101.

[37] Elmagarmid, A. K., P. G. Ipeirotis, and V. S. Verykios (2007) "Duplicate Record Detection: A Survey," *IEEE Trans. Knowl. Data Eng.*, **19**(1), pp. 1–16.

[38] On, B.-W., N. Koudas, D. Lee, and D. Srivastava (2007) "Group Linkage," in *Proceedings of the 23rd International Conference on Data Engineering*, pp. 496–505.

[39] sik Kim, H. and D. Lee (2007) "Parallel Linkage," in *Proceedings of the 16th ACM Conference on Information and Knowledge Management*, pp. 283–292.

[40] Hong, Y., T. Yang, J. Kang, and D. Lee (2008) "Record Linkage as DNA Sequence Alignment Problem," in *Proceedings of the International Workshop on Quality in Databases and Management of Uncertain Data*, pp. 13–22.

[41] pong Chan, K. and A. W.-C. Fu (1999) "Efficient Time Series Matching by Wavelets," in *Proceedings of the 15th International Conference on Data Engineering*, pp. 126–133.

[42] Keogh, E. J., K. Chakrabarti, M. J. Pazzani, and S. Mehrotra (2001) "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases," *Knowl. Inf. Syst.*, **3**(3), pp. 263–286.

[43] Lin, J., E. J. Keogh, L. Wei, and S. Lonardi (2007) "Experiencing SAX: a novel symbolic representation of time series," *Data Min. Knowl. Discov.*, **15**(2), pp. 107–144.

[44] Berndt, D. J. and J. Clifford (1994) "Using dynamic time warping to find patterns in time series," in *Proceedings of the AAAI Workshop on Knowledge Discovery in Databases*, pp. 359–370.

[45] Vlachos, M., D. Gunopulos, and G. Kollios (2002) "Discovering similar multidimensional trajectories," in *Proceedings of the 18th International Conference on Data Engineering*, pp. 673–684.

[46] Ding, H., G. Trajcevski, P. Scheuermann, X. Wang, and E. J. Keogh (2008) "Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures," *PVLDB*, **1**(2), pp. 1542–1552.

[47] Ratanamahatana, C. A. and E. J. Keogh (2005) "Three Myths about Dynamic Time Warping," in *Proceedings of the 2005 SIAM International Conference on Data Mining*, pp. 506–510.

[48] Hofmann, T. (1999) "Probabilistic Latent Semantic Analysis," in *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pp. 289–296.

[49] Wallach, H. (2006) "Topic Modeling: Beyond Bag-of-words," in *Proceedings of the 23rd International Conference on Machine Learning*, pp. 977–984.

[50] Wang, X., A. McCallum, and X. Wei (2007) "Topical N-grams: Phrase and Topic Discovery, with an Application to Information Retrieval," in *Proceedings of the 7th IEEE International Conference on Data Mining*, pp. 697–702.

[51] Gruber, A., Y. Weiss, and M. Rosen-Zvi (2007) "Hidden Topic Markov Models," in *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, pp. 163–170.

[52] Blei, D. M. and J. D. Lafferty (2007) "A Correlated Topic Model of Science," *Annals of Applied Statistics*, **1**.

[53] Griffiths, T. L., M. Steyvers, D. M. Blei, and J. B. Tenenbaum (2004) "Integrating Topics and Syntax," in *Proceedings of the 18th Annual Conference on Neural Information Processing Systems Conference*.

[54] Chemudugunta, C., P. Smyth, and M. Steyvers (2006) "Modeling General and Specific Aspects of Documents with a Probabilistic Topic Model," in *Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems*, pp. 241–248.

[55] Blei, D. M. and P. J. Moreno (2001) "Topic Segmentation with an Aspect Hidden Markov Model," in *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 343–348.

[56] Purver, M., K. P. Körding, T. L. Griffiths, and J. B. Tenenbaum (2006) "Unsupervised Topic Modelling for Multi-Party Spoken Discourse," in *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics.*

[57] Eisenstein, J. and R. Barzilay (2008) "Bayesian Unsupervised Topic Segmentation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 334–343.

[58] Eisenstein, J. (2009) "Hierarchical Text Segmentation from Multi-Scale Lexical Cohesion," in *Proceedings of the Conference of the North American Chapter of the Association of Computational Linguistics*, pp. 353–361.

[59] Shafiei, M. M. and E. E. Milios (2008) "A Statistical Model for Topic Segmentation and Clustering," in *Proceedings of the 21st Conference of the Canadian Society for Computational Studies of Intelligence*, pp. 283–295.

[60] Sriram, B., D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas (2010) "Short text classification in twitter to improve information filtering," in *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 841–842.

[61] Phan, X. H., M. L. Nguyen, and S. Horiguchi (2008) "Learning to classify short and sparse text & web with hidden topics from large-scale data collections," in *Proceedings of the 17th International Conference on World Wide Web*, pp. 91–100.

[62] Sun, A. (2012) "Short text classification using very few words," in *Proceeding of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1145–1146.

[63] Nishida, K., T. Hoshide, and K. Fujimura (2012) "Improving tweet stream classification by detecting changes in word probability," in *Proceedings of the 35th International ACM SIGIR conference on research and development in Information Retrieval*, pp. 971–980.

[64] Burger, J. D., J. C. Henderson, G. Kim, and G. Zarrella (2011) "Discriminating Gender on Twitter," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1301–1309.

[65] Bifet, A. and E. Frank (2010) "Sentiment Knowledge Discovery in Twitter Streaming Data," in *Proceedings of the 13th International Conference on Discovery Science*, pp. 1–15.

[66] Cheng, Z., J. Caverlee, and K. Lee (2010) "You are where you tweet: a content-based approach to geo-locating twitter users," in *Proceedings of the 19th ACM Conference on Information and Knowledge Management*, pp. 759–768.

[67] Cheng, Z., J. Caverlee, K. Lee, and D. Z. Sui (2011) "Exploring Millions of Footprints in Location Sharing Services," in *Proceedings of the 5th International Conference on Weblogs and Social Media*.

[68] Chang, H.-W., D. Lee, M. Eltaher, and J. Lee (2012) "@Phillies Tweeting from Philly? Predicting Twitter User Locations with Spatial Word Usage," in *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 111–118.

[69] Benevenuto, F., G. Magno, T. Rodrigues, and V. Almeida (2010) "Detecting Spammers on Twitter," in *Proceedings of the 7th annual Collaboration, Electronic messaging, AntiAbuse and Spam Conference*.

[70] Radinsky, K., E. Agichtein, E. Gabrilovich, and S. Markovitch (2011) "A word at a time: computing word relatedness using temporal semantic analysis," in *Proceedings of the 20th International Conference on World Wide Web*, pp. 337–346.

[71] Hu, J., H.-J. Zeng, H. Li, C. Niu, and Z. Chen (2007) "Demographic prediction based on user's browsing behavior," in *Proceedings of the 16th International Conference on World Wide Web*, pp. 151–160.

[72] Goel, S., J. M. Hofman, and M. I. Sirer (2012) "Who does what on the Web: Studying Web browsing behavior at scale," in *Proceedings of the 6th International AAAI Conference on Weblogs and Social Media*.

[73] Kosinski, M., P. Kohli, D. Stillwell, Y. Bachrach, and T. Graepel (2012) "Personality and website choice," in *Proceedings of the ACM Web Science Conference*.

[74] Quercia, D., R. Lambiotte, D. Stillwell, M. Kosinski, and J. Crowcroft (2012) "The Personality of popular Facebook users," in *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pp. 955–964.

[75] Bachrach, Y., M. Kosinski, T. Graepel, P. Kohli, and D. Stillwell (2012) "Personality and patterns of Facebook usage," in *Proceedings of the ACM Web Science Conference*.

[76] Quercia, D., M. Kosinski, D. Stillwell, and J. Crowcroft (2011) "Our Twitter profiles, ourselves: Predicting personality with Twitter," in *Proceedings of the 3rd IEEE International Conference on Social Computing*, pp. 180–185.

[77] Golbeck, J., C. Robles, M. Edmondson, and K. Turner (2011) "Predicting personality from Twitter," in *Proceedings of the 3rd IEEE International Conference on Social Computing*, pp. 149–156.

[78] Yang, T. and D. Lee (2009) "T3: On Mapping Text To Time Series," in *Proceedings of the 3rd Alberto Mendelzon International Workshop on Foundations of Data Management*.

[79] Salton, G., A. Wong, and C. S. Yang (1975) "A Vector Space Model for Automatic Indexing," *Commun. ACM*, **18**(11), pp. 613–620.

[80] Ponte, J. M. and W. B. Croft (1998) "A Language Modeling Approach to Information Retrieval," in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 275–281.

[81] Yang, T. and D. Lee (2013) "On handling textual errors in latent document modeling," in *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management*, pp. 2089–2098.

[82] Porteous, I., D. Newman, A. T. Ihler, A. U. Asuncion, P. Smyth, and M. Welling (2008) "Fast Collapsed Gibbs Sampling for Latent Dirichlet Allocation," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 569–577.

[83] Yang, T. and D. Lee (2011) "Towards noise-resilient document modeling," in *Proceedings of the 20th ACM Conference on Information and Knowledge Management*, pp. 2345–2348.

[84] Yang, T., D. Lee, and S. Yan (2013) "Steeler Nation, 12th Man, and Boo Birds: Classifying Twitter User Interests using Time Series," in *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 684–691.

[85] Manning, C. D. and H. Schutze (1999) *Foundations of statistical natural language processing*, MIT Press.

[86] Cristianini, N. and J. Shawe-Taylor (2000) *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press.

[87] Xi, X., E. J. Keogh, C. R. Shelton, L. Wei, and C. A. Ratanamahatana (2006) “Fast time series classification using numerosity reduction,” in *Proceedings of the 23rd International Conference on Machine Learning*, pp. 1033–1040.

[88] Bay, S. D. (1998) “Combining nearest neighbor classifiers through multiple feature subsets,” in *Proceedings of the 15th International Conference on Machine Learning*, pp. 37–45.

[89] Berndt, D. J. and J. Clifford (1994) “Using dynamic time warping to find patterns in time series,” in *Proceedings of the AAAI Workshop on Knowledge Discovery in Databases*, pp. 359–370.

[90] Shieh, J. and E. J. Keogh (2009) “*i*SAX: indexing and mining terabyte sized time series,” *Data Min. Knowl. Discov.*, **19**(1), pp. 24–57.

[91] Conover, M., J. Ratkiewicz, M. Francisco, B. Gonçalves, F. Menczer, and A. Flammini (2011) “Political Polarization on Twitter,” in *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*.

[92] Ding, C. H. Q. and X. He (2004) “K-means Clustering via Principal Component Analysis,” in *Proceedings of the 21st International Conference on Machine Learning*, p. 29.

Vita

Tao Yang

Tao Yang was born in Anqing, Anhui, China. He received his B.S degree in Computer Science from University of Science and Technology of China, and his M.S degree in Statistics from The Pennsylvania State University. He is a Ph.D. candidate and a graduate researcher in the College of Information Sciences and Technology at The Pennsylvania State University. His current research focuses on data mining, machine learning, social network analysis and statistical genetics.