The Pennsylvania State University
The Graduate School

IMPROVING PREDICTION, RECOMMENDATION, AND CLASSIFICATION USING USER-GENERATED CONTENT AND RELATIONSHIPS

A Dissertation in
Computer Science and Engineering
by
Hau-Wen Chang

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2015

The dissertation of Hau-Wen Chang was reviewed and approved[*] by the following:

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

Wang-Chien Lee
Associate Professor of Computer Science and Engineering
Co-Chair of Committee

Kamesh Madduri
Assistant Professor of Computer Science and Engineering

Dinghao Wu
Assistant Professor of Information Sciences and Technology

Lee Coraor
Associate Professor of Computer Science and Engineering
Director of Academic Affairs

[*]Signatures are on file in the Graduate School.

Abstract

In the dominance of social networks era, vast information is created and shared across the world each day. The uniqueness and the prevalence of these user-generated content present both challenges and opportunities. In this thesis, in particular, we study several tasks on mining the user-generated content with regard to textual content and link-based content.

First, we study the home location estimation for Twitter users from their shared textual content. We employ Gaussian Mixture Model to compensate the drawback in the Maximum Likelihood Estimation. We propose two unsupervised feature selection methods based on the notions of Non-Localness and Geometric-Localness to prune noisy data in the content.

Second, we study the item recommendation problem with a broader view of a social network system. By taking various relationships into consideration, the data sparseness problem common in recommendation tasks are alleviated. Based on the same characteristics principle, we propose a matrix co-factorization framework with a shared latent space to optimize the recommendation collectively. Several algorithms are proposed under the framework to exploit intricate relationships in a social network system.

Finally, we investigate the effectiveness of classification with the imperfect textual content extracted from videos, where often very limited information is available. Through means of automatic recognition techniques, some link-based content is enriched with a trade-off of incorrectness. We also propose a heuristics-based method to extract n-gram keyphrases from noisy textual content by taking the importance of sub-term keywords into consideration.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Billions of users are drawn to social networks to share their ideas or activities, and to learn information shared by others. As a result, a vast amount of information is generated in todays social networks. The sharing can take in many forms such as words, images, videos, or even links to other users showing the friendships. In general, user-generated content on social networks can be studied from two aspects: textual content and link-based content. Ideas expressed in written words in a particular language are considered as textual content, such as posts, messages, comments. Link-based content refers to the relationship between two objects, such as a user and a video, a user and another user and so on. We discuss challenges in mining the user-generated content in both types, and present our studies on specific mining tasks with regard to different types. Fig 1.1 gives an overview of this study.

1.1.1 Improving Prediction Using User-Generated Texts

Social network sites have gained much popularity in recent years. For example, 300 million users post 200 million of messages or tweets called tweets in 2011 on the leading Microblogging service provider, Twitter. The ever-growing new data from Twitter or Facebook presents opportunities to study social network users, their communication,

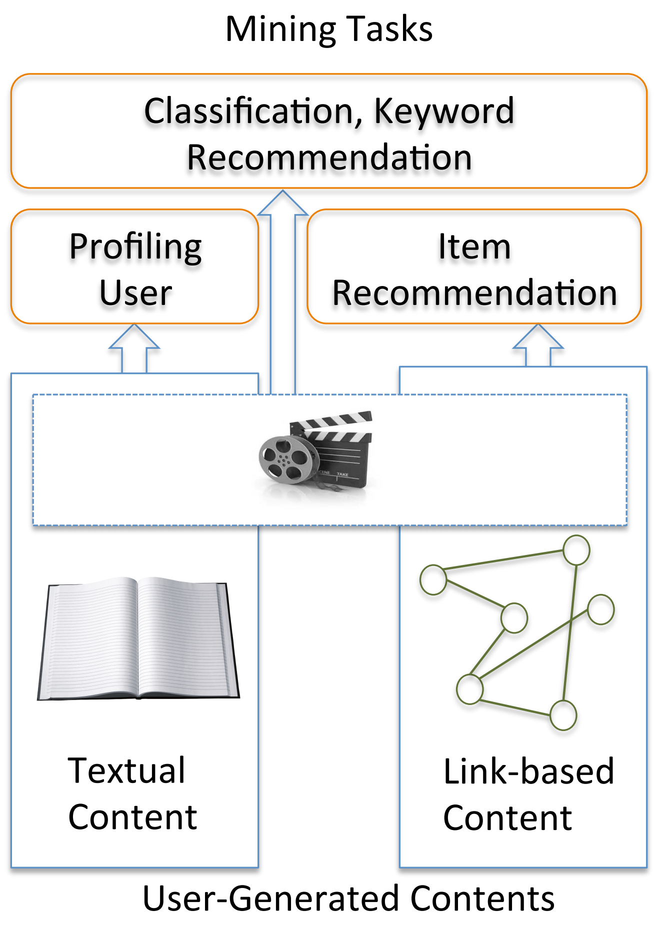


Figure 1.1. Overview of Thesis Study

and opinions. On the other hand, the new type of content also exhibits several distinct characteristics as compared with the formal textual corpus researchers used to study such as scientific articles, news reports and so on.

Firstly, the messages are short and many. For example, only 140 characters are allowed in a single tweet, making it difficult for topic detection and structural analysis beyond sentence. From observation, many messages relate to users daily activities and personal comments. For example, what they do, where they go, how they feel for an event or a product. Secondly, the messages are noisy. Users tend to be creative under the length limitation. As a result, abbreviations are often used while punctuations are often omitted. Self-created words, slangs, jagons, expressions (e.g. LOL) and emoticons (e.g. :-)) are popular while multi-lingual usage between messages is not uncommon. Grammatical analysis is therefore difficult to be effective. Finally, a message may have social network aspect. Messages such retweets, relaying messages from others to ones followers, reflect the relationships between two users. Despite the challenges, many promising applications have been proposed by taking advantage of the messages. For example, tweets have been analyzed to track flu epidemic[1], to localize illness by geographic region and to analyze symptoms medication usage[2], to capture breaking news[3], to detect earthquake[4], to verify rumor[5], and so on.

Geography plays an important role in our daily life, which in turn may have impacts on the content our messages. For example, users often tweet to plan a gathering on a nearby restaurant, to cheer a local sports team or to discuss local candidates in the election. All of which give a hint on the users whereabouts. The understanding of user location is beneficial in many applications, such as connecting the users in the neighborhood or recommending nearby restaurants. Although Twitter allows users to specify their location in their profile, users often either do not provide such geographic information (for laziness or privacy concern) or provides them only in inconsistent granularities (e.g., country, state, or city) and reliability. Therefore, being able to automatically uncover user's home location using her social media data becomes an important problem.

1.1.2 Improving Recommendation Using User-Generated Relationships

A social network itself is a natural link-based content, connecting users to users. This form of content is very prevalent as we extend the connection to the various objects on the Web. For example, online video sharing social networks such as Youtube links users to videos, as well as users. Flickr and Instagram are among popular social networks linking users to images. LinkedIn connects users through communities or organizations. Mining tasks on link-based content tend to more difficult text mining, since links are much less informative than the textual content. Although some linked objects might have textual information associate with them such as metadata, but often they are very limited and might be expensive to acquire. Therefore, many techniques developed for textual-content is not always applicable.

One of the fundamental tasks in studying this type of data is to make a recommendation for users, and even for various types of objects, such as recommendation a potential member for a community, or a video to a subscripted channel or a collection. Current studies for item recommendation with the link-based content often focus on individual task, and overlook the intricate connections across tasks. Often, we can find a correlation across different types of links on the same social network platform. For example, a user shared an animal image is more likely to tag an animal image from others as one of her favorite, or to join a group of animal photos. Similarly, we could also recommend possible friends to a user, if the two share similar interests. In general, we could view a social network as a system accommodating entities of various types such as users, groups and images. We then differentiate the connections between entities of the same type from those between entities of different types, referred to as self-relationship and inter-relationship, respectively. Each type of relationships presents a unique recommendation task. By taking various relationships into consideration, we might enrich the data and alleviate data sparseness problem, which is generally considered as one of the major obstacles in making quality recommendation. In this work, we first investigate the correlation between various relationships. From the analysis, we propose the same characteristics principle, which leads to a co-factorization framework where entities of the same type has a shared latent space in a given system. We then propose several algorithms under the framework to take advantages of different relationships. We show that by considering multiple tasks collectively, the recommendation can be further improved.

1.1.3 Improving Classification Using User Generated Videos

One of the challenges in mining link-based content is the data is less informative than the the textual content. With the advance of the automatic recognition techniques, the textual content can be uncovered partially on some object in the link-based content, such as images with texts or videos with speeches. The videos with speeches, also known as spoken documents, is a popular forms for modern users to broadcast their ideas, and well shared in many multimedia platforms, such as Youtube and Vimeos. Education oriented websites such as Coursera and TED also create a good number of spoken documents that are often popularly shared across social networks. The textual content extracted from the automatic recognition process tends to provide richer informative than we can studied through its links, or even than the metadata associate with it. However, the qualities of extracted content may vary, depending on many variables, such the recording condition, speaking speed, the training process, and so on.

In this study, we first study the effectiveness of the text classification with spoken documents. Using both real and synthesized data, we compare the performance of different text categorization methods, and learn the impacts of various factors with the transcripts such as quality and length. To further take advantages of the imperfect content, we propose a heuristic method to extracts keywords and keyphrases for a spoken document by taking the importance of sub-terms into consideration. We show that the proposed method works well with noisy spoken documents from various knowledge domains, and extracts keyphrases better than existing tf-idf extraction method does.

1.2 Thesis Organization

The rest of this thesis is organized as follows.

In Chapter 2, we present our method for estimating Twitter user's home location. We propose a probability framework and two novel local word selections to further improve the performance. We compare our results to other state-of-the-art methods.

In Chapter 3, we our study on improve item recommendation for link-based content. We propose several algorithms that take multiple recommendation tasks into consideration, and collectively, make better recommendation.

In Chapter 4, we investigate capability of the noisy transcripts on the classification task. We present a performance comparison of classifying videos by different types of transcript. We also propose a method to automatic generate keywords for videos from their noisy transcripts, when metadata is not available.

In Chapter 5, we conclude the current work and discuss the possible directions for future research.

Chapter 2

Improving Prediction Using User-Generated Texts

2.1 Introduction

Knowing users' home locations in social network systems bears an importance in applications such as location-based marketing and personalization. In many social network sites, users can specify their home locations along with other demographics information. However, often, users either do not provide such geographic information (for laziness or privacy concern) or provide them only in inconsistent granularities (e.g., country, state, or city) and reliabilities.

Intuition behind the problem is that geography plays an important role in our daily lives so that word usage patterns in Twitter may exhibit some geographical clues. For example, users often tweet about a local shopping mall where they plan to hang out, cheer a player in local sports team, or discuss local candidates in elections. Therefore, it is natural to take this observation into consideration for location estimation.

2.1.1 Problem Definition

In this study, we focus on the case of Twitter users and try to predict their city locations based on only the *contents* of their tweet messages, without using other information such as user profile metadata or network features. When such additional information is available, we believe one can estimate user locations with a better accuracy and will leave it as a future work. Our problem is formalized as follows:

- For a user u, given a set of his/her tweet messages $T_u = \{t_1, ..., t_{|T_u|}\}$, where t_i is a tweet message up to 140 characters, and a list of candidate cities, C, predict a city c $(\in C)$ that is most likely to be the home location of u.

2.2 Related Work

The location estimation problem which is also known as *geolocating* or *georeferencing* has gained much interests recently. Table2.1 shows a summary of related studies. One of pioneering work and state-of-the-art is conducted by Cheng et al.[6] which uses a proposed probabilistic framework to estimate city-level location based only on the contents of tweets without considering other geospatial clues. The performance is greatly improved with porposed feature selection, which requires a manual selection of local words for training a classification model. Chandra et al. [7] extended Cheng's work by taking the "reply-tweet" relation into consideration in addition to the text contents. The results, however, is not better than Cheng's work. Hecht et al [8] analyzed the user location filed in user profile and used Multinomial Naive Bayes to estimate user's location in state and country level. They also observed that 34% of Twitter user do not provide the location information in their profile. [9] approached the problem with a language model approach with varying levels of granularities, from zip codes to country levels.

Studies have also been conducted by utilizing social relation in addition of the textual contents. Backstrom et al.[10] proposed an algorithm that predicts the location of an individual from a sparse set of located Facebook friends with performance that exceeds IP-based geolocation. Similarly, Flap[11] predicted Twitter user's mobility by treating users with known GPS positions as noisy sensors of the location of their friends. Other location related studies include [12, 13]. [12] studied the problem of matching a tweet to a list of objects of a given domain (e.g., restaurants) whose geolocation is known. Their study assumes that the probability of a user tweeting about an object depends on the distance between the user's and the object's locations. The matching of tweets in turn helps decide the user's location. [13] studied the problem of associating a single tweet to a tag of point of interests, e.g., club, or park, instead of user's home location.

A similar task that has been extensively studied is to predict the origin of multimedia data, e.g. where a photo is taken, or a video is recorded, with its associated tags,

Table 2.1. Summary of related studies on location estimation.

Paper	Task	Features	Method	Dataset
[9]	User location	Text	Language model	Spritzer, Firehose
[6]	User location	Text	Proposed, feature selection, smoothing	Cheng
[7]	User location	Text, hashtag, retweet	Proposed	Cheng
[8]	User location	Text	Multinomial Naive Bayes	Hechet
[10]	User location	Friendship	Proposed	Backstrom
[11]	User mobility	Friendship	Proposed	Sadilek
[14]	Image location	User tag	Language model, smoothing	Serdyukov
[15]	Image location	User tag	Language model, feature selection	PlacingTask11
[16]	Image location	User tag, visual feature	Proposed	PlacingTask10

description, images features, or audio features. Serdyukov et al.[14] used a language model approach to predict the image location with the tags defined by user. Several smoothing techniques are proposed to improve the accuracy. Van Laere et al.[15] used a language model approach and combination of several classifiers by Dempster-Shafer theory to improve the performance. Kelm et al.[16] use fusion models built from different features to approach problem. The placing task in MediaEval provides a benchmark evaluation for geolocating photos and videos on Flickr[17].

2.3 Proposed Methods

Recently, the *generative methods* (e.g., [14, 9, 15]) have been proposed to solve the proposed Problem 1. Assuming that each tweet and each word in a tweet is generated independently, the prediction of home city of user u given his or her tweet messages is made by the conditional probability under Bayes's rule and further approximated by ignoring $P(T_u)$ that does not affect the final ranking as follows:

$$P(C|T_u) = \frac{P(T_u|C)P(C)}{P(T_u)}$$

$$\propto \; P(C) \prod_{t_j \in T_u} \prod_{w_i \in t_j} P(w_i|C)$$

where w_i is a word is a tweet t_j. If $P(C)$ is estimated with the maximum likelihood, the cities having a high usage of tweets are likely to be favored. Another way is to assume a uniform prior distribution among cities, also known as the *language model* approach in IR, where each city has its own language model estimated from tweet messages. For a user whose location in unknown, then, one calculates the probabilities of the tweeted words generated by each city's language model. The city whose model generates the highest probability of the tweets from the user is finally predicted as the home location. This approach characterizes the language usage variations over cities, assuming that users have similar language usage within a given city. Assuming a uniform $P(C)$, we propose another approach by applying Bayes rule to the $P(w_i|C)$ of above formula and replace the products of probabilities by the sums of log probabilities, as is common in probabilistic applications:

$$\begin{aligned} P(C|T_u) &\propto P(C) \prod_{t_j \in T_u} \prod_{w_i \in t_j} \frac{P(C|w_i)P(w_i)}{P(C)} \\ &\propto \sum_{t_j \in T_u} \sum_{w_i \in t_j} \log(P(C|w_i)P(w_i) \end{aligned}$$

Therefore, given C and T_u, the home location of the user u is the city c $(\in C)$ that maximizes the above function as:

$$\text{argmax}_{c \in C} \sum_{t_j \in T_u} \sum_{w_i \in t_j} \log(P(c|w_i)P(w_i)$$

Instead of estimating a language model for a city, this model suggests to estimate the city distribution on the use of each word, $P(C|w_i)$, which we refer to it as **spatial word usage** in this study, and aggregate all evidences to make the final prediction. Therefore, its capability critically depends on whether or not there is a distinct pattern of word usage among cities. Note that the proposed model is similar to the one used in [6], $P(C|T_u) \propto \sum_{t_j \in T_u} \sum_{w_i \in t_j} P(C|w_i)P(w_i)$, where the design was based on the observation rather than derived theoretically.

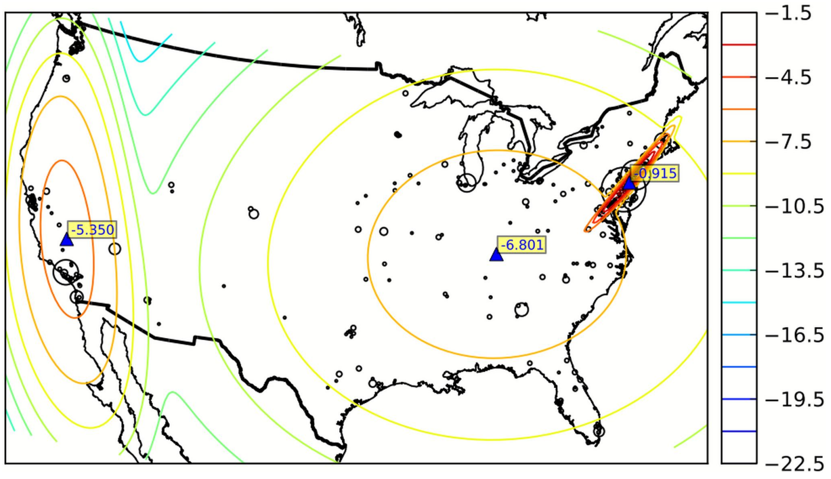
The *Maximum Likelihood Estimation (MLE)* is a common way to estimate $P(w|C)$

and $P(C|w)$. However, it suffers from the data sparseness problem that underestimates the probabilities of words of low or zero frequency. Various *smoothing* techniques such as Dirichlet and Absolute Discount [18] are proposed. In general, they distribute the probabilities of words of nonzero frequency to the words of zero frequency. For estimating $P(C|w)$, the probability of tweeting a word in locations where there are zero or few twitter users are likely to be underestimated as well. In addition to these smoothing techniques, some probability of a location can be distributed to its neighboring locations, assuming that two neighboring locations tend to have similar word usages. While reported effective in other IR applications, however, the improvements from such smoothing methods to estimate user locations have been shown to be limited in the previous studies [14, 6]. One of our goals in this study is therefore to propose a better estimation for $P(C|w)$ to improve the prediction while addressing the spareness problem.

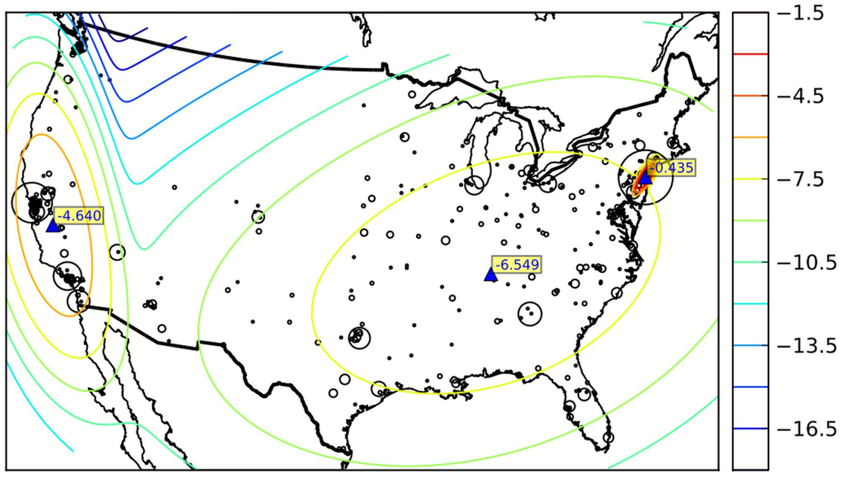
2.3.1 Estimation with Gaussian Mixture Model

The Backstrom model [19] demonstrated that users in a particular location tend to query some search keywords more often than users in other locations, especially, for some topic words such as sport teams, city names, or newspaper. For example, as demonstrated in [19], `redsox` is searched more often in New England area than other places. In their study, the likelihood of a keyword queried in a given place is estimated by $Sd^{-\alpha}$, where S indicates the strength of frequency on the local center of the query, and α indicates the speed of decreasing when the place is d away from the center. Therefore, the larger S and α in the model of a keyword shows a higher local interest, indicating strong local phenomena. While promising results are shown in their analysis with query logs, however, this model is designed to identify the center rather than to estimate the probability of spatial word usage and is difficult to handle the cases where a word exhibits multiple centers (e.g., `giants` for the NFL NY Giants and the MLB SF Giants).

Therefore, to address such issues, we propose to use the bivariate *Gaussian Mixture Model (GMM)* as an alternative to model the spatial word usage and to estimate $P(C|w)$. GMM is a mature and widely used technique for clustering, classification, and density estimation. It is a probability density function of a weighted sum of a number of Gaussian components. Under this GMM model, we assume that each word has a number of centers of interests where users tweet it more extensively than users in other locations,



(a) `phillies`

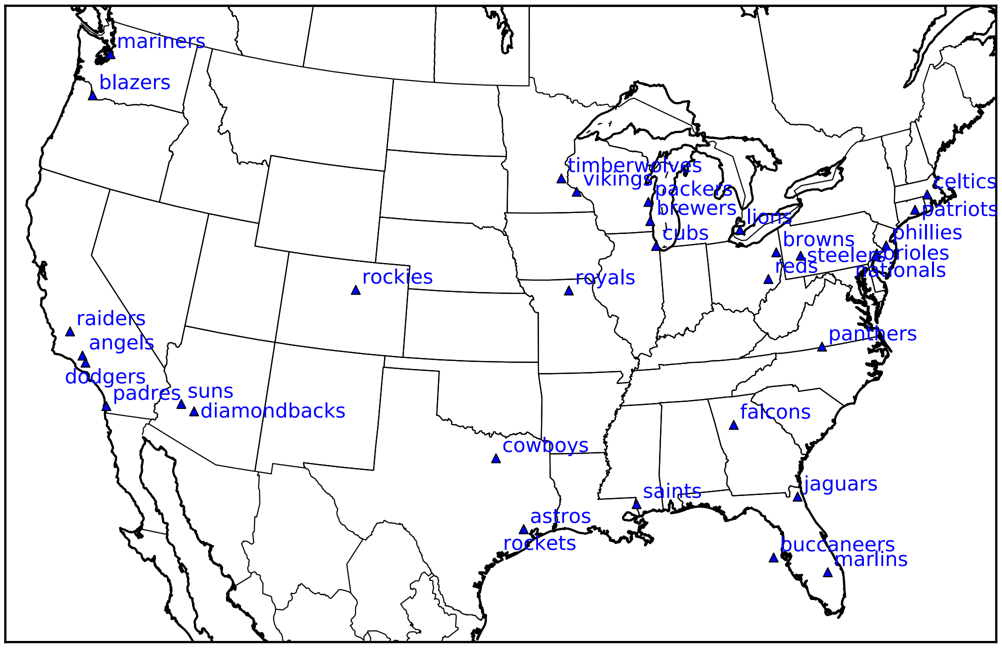


(b) `giants`

Figure 2.1. Results of GMM estimation on selected words in Twitter data set.

thus having a higher $P(c|w)$, and that the probability of a user in a given location tweeting a word is influenced by the word's multiple centers, the magnitudes of the centers, and user's geographic distances to those centers. Formally, using GMM, the probability of a city c on tweeting a word w is:

$$P(c|w) = \sum_{i=1}^{K} \pi_i N(c|\mu_i, \Sigma_i)$$



US sports teams

Figure 2.2. US sports teams home location with GMM estimation.

where each $N(c|\mu_i, \Sigma_i)$ is a bivariate Gaussian distribution with the density as:

$$\frac{1}{2\pi|\Sigma_i|^{1/2}} \exp\left\{-\frac{1}{2}(c-\mu_i)^T\Sigma_i^{-1}(c-\mu_i)\right\}$$

where K is the number of components and $\sum_{i=1}^{K}\pi_i = 1$. To estimate $P(C|w)$ with GMM, each occurrence of the word w is seen as a data point (lon, lat), the coordinate of the location where the word is tweeted. In other words, if a user has tweeted `phillies` 3 times, there are 3 data points (i.e., (lon, lat)) of the user location in the data set to be estimated by GMM. Upon convergence, we compute the density for each city c in C, and assign it as the $P(c|w)$. In the GMM-estimated $P(C|w)$, the mean of a component is the hot spot (i.e., center) of tweeting the word w, while the covariance determines the magnitude of a center. Similar to the Backstrom model, the chance of tweeting a word w decreases exponentially away from the centers. Unlike the Backstrom model, however, GMM easily generalizes to multiple centers and considers the influences under different centers (i.e., components) altogether. Furthermore, GMM is computationally efficient since the underlying EM algorithm generally converges very quickly. Compared to MLE, GMM may yield a high probability on a location where there are few Twitter user, as long as the the location is close to a hot spot. It may also assign a low probability to locations with high frequency of tweeting a word if that location is far way from all the hot spots. On the other hand, GMM-based estimation can be also viewed as a radical geographic smoothing such that neighboring cities around the centers are favored.

In Fig. 2.1(a), we show the contour lines of log-likelihood of a GMM estimation with 3 components (i.e., $K = 3$) on the word `phillies` which has been tweeted 1,370 times from 229 cities in Twitter data set (see Section 2.4). A black circle in the map indicates a city, where radius is proportional to the frequency of `phillies` being tweeted by users in the city. The corresponding centers are plotted as blue triangles. Note that there is a highly concentrated cluster of density around the center in northeast, close to Philadelphia, which is the home city of `phillies`. The other two centers and their surrounding areas have more low and diluted densities. Note that GMM works well in clustering probabilities around the location of interests with the evidences of tweeting location, even if the number of components (K) is not set to the exact number of centers. Sometimes, there might be more than one distinct cluster in a city distribution for a word. For example, `giants` is a name of a NFL team (i.e., New York Giants) as well a MLB team (i.e., San Francisco Giants). Therefore, it is likely to be often mentioned by Twitter users from both cities. As shown in Fig. 2.1(b), the two highest peaks are close to both cities. The peak near New York city has a higher likelihood than that near San Francisco, indicating `giants` is a more popular topic for users around New York city area. In Fig. 2.2, finally, we show that GMM can be quite effective in identifying the location of interests by selecting the highest peaks for various sport teams in US.

As shown in above example, in fitting the spatial word usage with GMM, if a word has strong patterns, one or more major clusters are likely be formed and centered around the locations of interests with highly concentrated densities. If two close locations are both far away from the major clusters, their probabilities are likely to be smoothed out to a similar and low level, even if they are distinct in actual tweeted frequencies.

2.3.2 Unsupervised Selection of Local Words

[6] made an insightful finding that in estimating locations of Twitter users, using only a selected set of words that show strong locality (termed as *local words*) instead of using entire corpus can improve the accuracy significantly (e.g., from 0.101 to 0.498).

Similarly, we assumed that words have some locations of interests where users tend to tweet extensively. However, *not* all words have a strong pattern. For example, if a user tweets `phillies` and `libertybell` frequently, the probability for Philadelphia to be her home location is likely to be high. On the other hand, even if a user tweets words like `restaurant` or `downtown` often, it is hard to associate her with a specific location. That is because such words are commonly used and their usage will not be restricted locally. Therefore, excluding such globally occurring words would likely to improve overall performance of the task.

In particular, in selecting local words from the corpus, [6] used a supervised classification method. They manually labeled around 19,178 words in a dictionary as either local or non-local and used parameters (e.g., S, α) from the Backstorm's model and the frequency of a word as features to build a supervised classifier. The classifier then determines whether other words in the data set are local. Despite the promising results, we believe that such a *supervised selection approach is problematic*–i.e., not only their labeling process to manually create a ground truth is labor intensive and subject to human bias, it is hard to transfer labeled words to new domain or data set. Moreover, the dictionary used in labeling process might not differentiate the evidences on different forms of a word. For example, the word `bears` (i.e., name of an NFL team) is likely to be a local word, while the word `bear` might not be. As a result, we believe that a better approach is to automate the process (i.e., *unsupervision*) such that the decision on the localness of a word is made only by their actual spatial word usage, rather than their semantic meaning being interpreted by human labelers. Toward this challenge, in the following, we propose two *unsupervised* methods to select a set of "local words" from a corpus using the evidences from tweets and their tweeter locations directly.

2.3.3 Finding Local Words by Non-Localness: NL

Stop words such as `the`, `you`, or `for` are in general commonly used words that bear little significance and considered as noises in many IR applications such as search engine or text mining. For instance, compare Fig. 2.3 showing the frequency distribution for the stop word `for` to Fig. 2.1 showing that for word with strong local usage pattern like `giants`. In Fig. 2.3, one is hard to pinpoint a few hotspot locations for `for` since it is globally used.In the location prediction task, as such, the spatial word usage of these

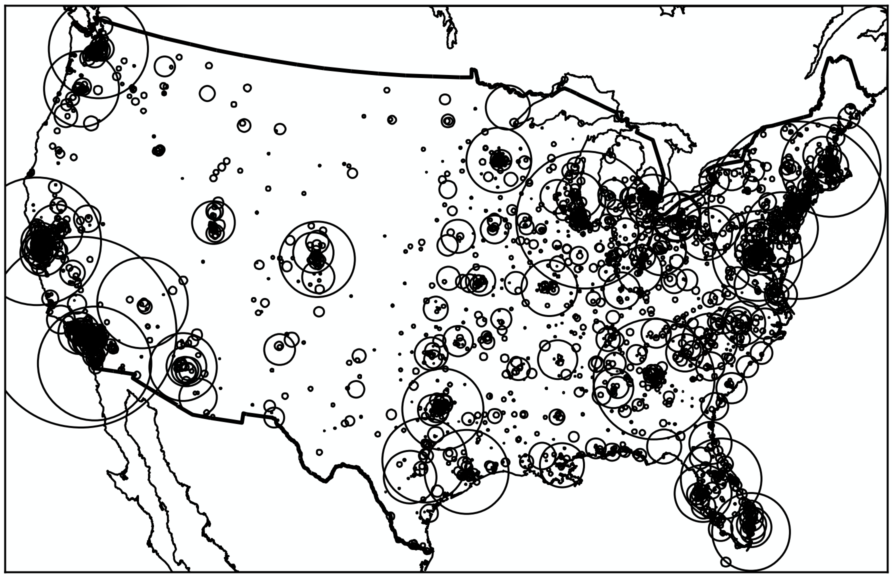


Figure 2.3. The occurrences of the stop word `for` in Twitter data set.

stop words shows a somewhat ***uniform*** distributions adjusted to the sampled data set. As an automatic way to filter noisy non-local words out from the given corpus, therefore, we propose to use the stop words as *counter examples.* That is, local words tend to have the farthest distance in spatial word usage pattern to stop words. We first estimate a spatial word usage $p(C|w)$ for each word as well as stop words. The similarity of two words, w_i and w_j, can be measured by the distance between two probability distributions, $p(C|w_i)$ and $p(C|w_j)$. We consider two divergences for measuring the distance: Symmetric Kullback-Leibler divergence (sim_{SKL}) and Total Variation (sim_{TV}):

$$\begin{aligned}
\text{sim}_{SKL}(w_i, w_j) &= \sum_{c \in C} P(c|w_i) \ln \frac{P(c|w_i)}{P(c|w_j)} + P(c|w_j) \ln \frac{P(c|w_j)}{P(c|w_i)} \\
\text{sim}_{TV}(w_i, w_j) &= \sum_{c \in C} |P(c|w_i) - P(c|w_j)|
\end{aligned}$$

For a given stop word list $S = \{s_1, ..., s_{|S|}\}$, we then define the *Non-Localness*, $NL(w)$, of a word w as the average similarity of w to each stop word s in S, weighted by the number of occurrences of s (i.e., frequency of s, $freq(s)$) :

$$NL(w) = \sum_{s \in S} \text{sim}(w, s) \frac{freq(s)}{\sum\limits_{s' \in S} freq(s')}$$

From the initial tweet message corpus, finally, we can rank each word w_i by its $NL(w_i)$ score in *ascending* order and use top-k words as the final "local" words to be used in the prediction.

2.3.4 Finding Local Words by Geometric-Localness: GL

Intuitively, if a word w has: (1) a smaller number of cities with high probability scores (i.e., only a few peaks), and (2) smaller average inter-city geometric distances among those cities with high probability scores (i.e., geometrically clustered), then one can view w as a local word. That is, a local word should have a high probability density clustered within a small area. Therefore, based on these observations, we propose the *Geometric-Localness*, GL, of a word w:

$$GL(w) = \frac{\sum\limits_{c'_i \in C'} P(c'_i|w)}{|C'|^2 \frac{\sum \text{geo-dist}(c_u, c_v)}{|\{(c_u, c_v)\}|}}$$

where geo-dist(c_u, c_v) measures the geometric distance in miles between two cities c_u and c_v. Suppose one sort cities c $(\in C)$ according to $P(c|w)$. Using a user-set threshold parameter, r $(0 < r < 1)$, then, one can find a sub-list of cities $C' = (c'_1, ..., c'_{|C'|})$ s.t. $P(c'_i|w) \geq P(c'_{i+1}|w)$ and $\sum_{c'_i \in C'} P(c'_i|w) \geq r$. In the formula of $GL(w)$, the numerator then favors words with a few "peaky" cities whose aggregated probability scores satisfy the threshold r. The denominator in turn indicates that $GL(w)$ score is inversely proportional to the number of "peaky" cities (i.e., $|C'|^2$) and their average inter-distance (i.e., $\frac{\sum \text{geo-dist}(c_u, c_v)}{|\{(c_u, c_v)\}|}$). From the initial tweet message corpus, finally, we rank each word w_i by its $GL(w_i)$ score in *descending* order and use top-k words as the final "local" words to be used in the prediction.

2.4 Experimental Validation

2.4.1 Experimental Setup

For validating the proposed ideas, we used the same Twitter data set collected and used by [6]. This data set was originally collected between Sep. 2009 and Jan. 2010 by crawling through Twitter's public timeline API as well as crawling by breadth-first search through social edges to crawl each user's followees/followers. The data set is further split into *training* and *test* sets. The training set consists of users whose location is set in city levels and within the US continental, resulting in 130,689 users with 4,124,960 tweets. The test set consists of 5,119 active users with around 1,000 tweets from each, whose location is recorded as a coordinate (i.e., latitude and longitude) by GPS device, a much more trustworthy data than user-edited location information. In our experiments, we considered only 5,913 US cities with more than 5,000 of population in Census 2000 U.S. Gazetteer. Therefore, the problem that we experimented is to correctly predict *one* city out of 5,913 candidates as the home location of each Twitter user. We preprocess the training set by removing non-alphabetic characters (e.g., "@") and stop words, and selects the words of at least 50 occurrences, resulting in 26,998 unique terms at the end in our dictionary. No stemming is performed since singular and plural forms may provide different evidences as discussed in Section 2.3.2.

To measure the effectiveness of estimating user's home location, we used the following two metrics also used in the literature [6, 7, 14]. First, the *accuracy (ACC)* measures the average fraction of successful estimations for the given user set U:

$$ACC = \frac{|\{u|u \in U \text{ and } dist(Loc_{true}(u), Loc_{est}(u)) \le d\}|}{|U|}$$

The successful estimation is defined as when the distance of estimated and ground-truth locations is less than a threshold distance d. Like [6, 7], we use $d = 100$ (miles) as the threshold. Second, for understanding the overall margins of errors, we use the *average error distance (AED)* as:

$$AED = \frac{\sum_{u \in U} dist(Loc_{true}(u), Loc_{est}(u))}{|U|}$$

Table 2.2. Baseline results using different models.

Probability Model	ACC	AED
(1) $\sum\sum \log(P(c\|w_i)P(w_i))$	0.1045	1,760.4
(2) $\sum\sum P(c\|w_i)P(w_i)$	0.1022	1,768.73
(3) $\sum\sum \log P(w_i\|c)$	0.1914	1,321.42

Table 2.3. Results of Model (1) on GMM with varying # of components K.

K	1	2	3	4	5
ACC	0.0018	0.025	**0.3188**	0.2752	0.2758
AED	958.94	1785.79	**700.28**	828.71	826.1

K	6	7	8	9	10
ACC	0.2741	0.2747	0.2739	0.2876	0.3149
AED	830.62	829.14	830.33	786.34	746.75

2.4.2 Baselines

In Section 2, we compared three different models as discussed in Sec 2.3 to understand the impact of selecting the underlying probability frameworks. Table 2.2 presents the results of different models for location estimation. All the probabilities are estimated with MLE using all words in our dictionary. The baseline Models (1) and (2) (proposed by [6]) utilize the spatial word usage idea, and have around 0.1 of ACC and around 1,700 miles in AED. The Model (3), a language model approach, shows a much improved result–about two times higher ACC and AED with 400 miles less. These results are considered as baselines in our experiments.

2.4.3 Prediction with GMM Based Estimation

Next, we study the impact of the proposed GMM estimation[1] for estimating locations. In general, the results using GMM shows much improvements over baseline results of Table 2.2. Table 2.3 shows the results using Model (1) whose probabilities are estimated by GMM with different # of components K, using all the words in the corpus. Except the cases with $K = 1$ and $K = 2$, all GMM based estimations show substantial improvements over MLE based ones, where the best ACC (0.3188) and AED (700.28

[1]Using EM implementation from scikit-learn, `http://scikit-learn.org`

miles) are achieved at $K = 3$. Although the actual # of locations of interests varies for each word, in general, we believe that the words that have too many location of interests are unlikely to make contribution to the prediction. That is, as K becomes large, the probabilities are more likely to be distributed, thus making the prediction harder. Therefore, in subsequent experiments, we focus on GMM with a small # of components.

2.4.4 Prediction with Unsupervised Selection of Local Words

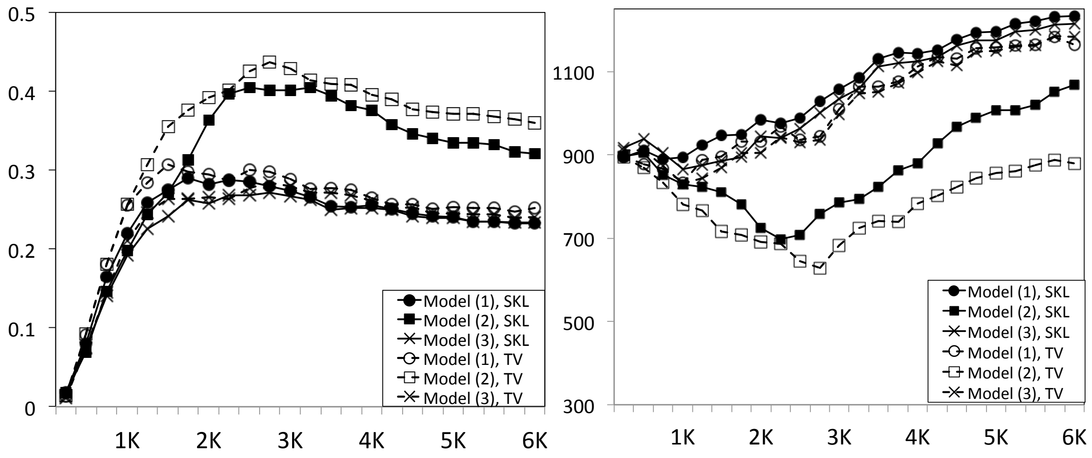
We attempt to see if the "local words" idea first proposed in [6] can be validated even when local words are selected in the unsupervised fashion (as opposed to [6]'s supervised approach). In particular, we validate with two unsupervised methods that we proposed on MLE estimation.

2.4.4.1 Non-Localness (NL)

In measuring $NL(w)$ score of a word w, we use the English stop word list from SMART system [20]. A total of 493 stop words (out of 574 in the original list), roughly 1.8% of all terms in our dictionary, occurred about 23M times (52%) in the training data. Due to their common uses in the corpus, such stop words are viewed as the least indicative of user locations. Therefore, $NL(w)$ measures the degree of similarity of w to average probability distributions of 493 stop words. Accordingly, if w shows the most dissimilar spatial usage pattern, i.e. $P(C|w)$, from those of stop words, then w is considered to be a candidate local word. The ACC and AED (in miles) results are shown in Fig. 2.4, as a function of the # of local words used (i.e., chosen as top-k when sorted by $NL(w)$ scores). In summary, Model (2) shows the best result of ACC (0.43) and AED (628 miles) with 3K local words used, a further improvement over the best result by GMM in Section 2.4.3 of ACC (0.3188) and AED (700.28 miles). Model (1) has a better ACC but a worse AED than Model (3) has. In particular, local words chosen using sim$_{TV}$ as the similarity measure outperforms sim$_{SKL}$ for all three Models.

2.4.4.2 Geometric-Localness (GL)

Our second approach selects a word w as a local word if w yields only a small number of cities with high probability scores (i.e., only a few peaks) and a smaller average inter-city geometric distances. Fig. 2.5(a) and (b) show the ACC and AED of three probability



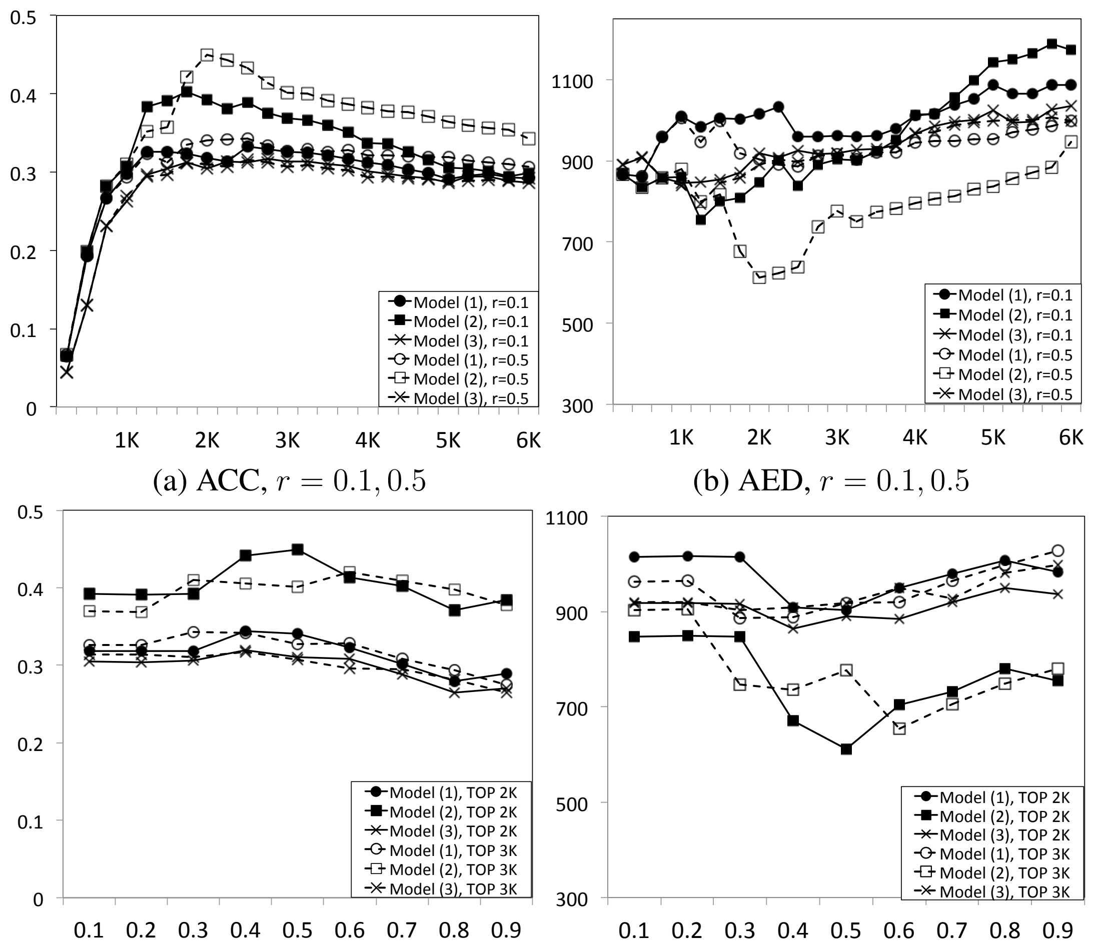
(a) ACC (b) AED

Figure 2.4. Results with local words selected by *Non-Localness* (NL) on MLE estimation (X-axis indicates # of top-k local words used).

models using either $r = 0.1$ and $r = 0.5$. The user-set parameter r (=$\sum_{c'_i \in C'} P(c'_i|w)$) of $GL(w)$ formula indicates the sum of probabilities of top candidate cities C'. Overall, all variations show similar behavior, but in general, Model (2) based variations outperform Model (1) or (3) based ones. Model (2) in particular achieves the best performance of ACC (0.44) and AED (600 miles) with $r = 0.5$ and 2K local words. Note that this is a further improvement over the previous case using NL as the automatic method to pick local words–ACC (0.43) and AED (628 miles) with 3K local words. Fig. 2.5(c) and (d) show the impact of r in $GL(w)$ formula, in X-axis, with the number of local words used fixed at 2K and 3K. In general, GL shows the best results when r is set to the range of 0.4 – 0.6. In particular, Model (2) is more sensitive to the choice of r than Models (1) and (3). In general, we found that GL slightly outperforms NL in both ACC and AED metrics.

2.4.5 Prediction with GMM and Local Words

In previous two sub-sections, we show that both GMM based estimation with all words and MLE based estimation with unsupervised local word selection are effective, compared to baselines. Here, further, we attempt to improve the result by combining both approaches to have unsupervised local word selection on the GMM based estimation. We first use the GMM to estimate $P(C|w)$ with $K = 3$, and calculate both $NL(w)$



(a) ACC, $r = 0.1, 0.5$ (b) AED, $r = 0.1, 0.5$

(c) ACC, # word=2K, 3K (d) AED, # word=2K, 3K

Figure 2.5. Results with local words selected by *Geometric-Localness* (GL) on MLE estimation (X-axis in (a) and (b) indicates # of top-k local words used and that in (c) and (d) indicates r of $GL(w)$ formula).

and $GL(w)$ using $P(C|w)$. Finally, we use the top-k local words and their $P(C|w)$ to predict user's location. Since Model (3) makes a prediction with $P(W|C)$ rather than $P(C|W)$, GMM based estimation cannot be used for Model (3), and thus is not compared. Due to the limitation of space, we report the best case using $NL(w)$ in Fig 2.6. Model (1) generally outperforms Model (2) and achieves the best result so far for both ACC (0.486) and AED (583.2 miles) with sim$_{TV}$ using 2K local words. While details are omitted, it is worthwhile to note that when used together with GMM, NL in general outperforms GL, unlike when used with MLE.

Table 2.4 illustrates examples where cities are predicted successfully by using NL-

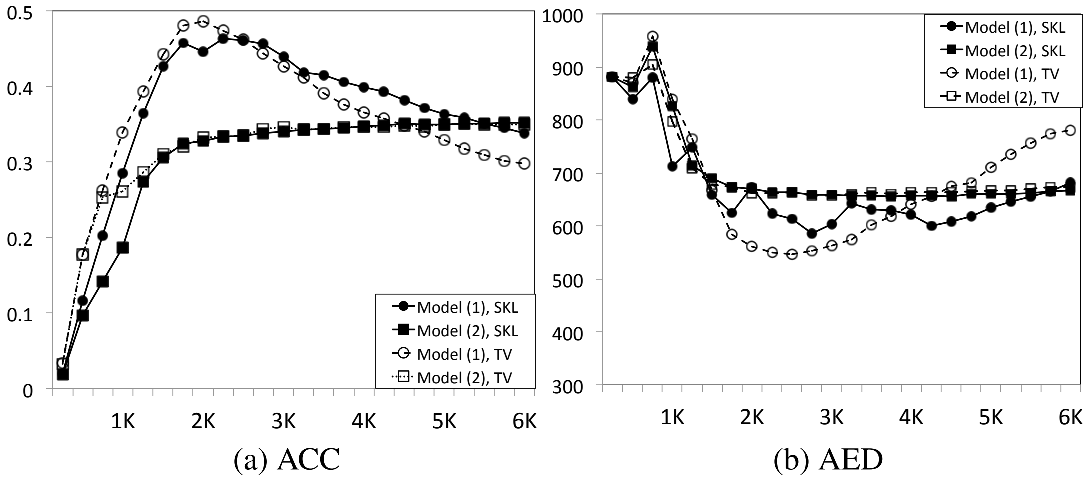


Figure 2.6. Results with local words selected by *Non-Localness* (NL) on GMM estimation (X-axis indicates # of top-k local words used).

Table 2.4. Examples of correctly estimated cities and corresponding tweet messages (local words are in bold face).

Est. City	Tweet Message
Los Angeles	i should be working on my monologue for my **audition** thursday but the thought of memorizing something right now is crazy
Los Angeles	i knew deep down inside ur powell s biggest fan p **lakers** will win again without **kobe** tonight haha if morisson leaves **lakers** that means elvan will not be rooting for **lakers** anymore
New York	the march **vogue** has caroline trentini in some awesome **givenchy** bangles i found a similar look for less an intern from teen **vogue** fashion dept just e mailed me asking if i needed an assistant aaadorable

selected local words and with GMM-based estimation. Note that words such as `audition` (i.e., the Hollywood area is known for movie industries) and `kobe` (i.e., name of the basketball player based in the area) are a good indicator of the city of the Twitter user.

In summary, overall, Model (1) shows a better performance with GMM while Model (2) with MLE as the estimation model. In addition, Model (1) usually uses less words to reach the best performance than Model (2) does. In terms of selecting local words, NL works better than GL in general, with sim$_{TV}$ in particular. In contrast, the best value of r depends on the model and the estimation method used. The best result for each model is summarized in Table 2.5 while further details on different combinations of those best

Table 2.5. Summary of best results of probability and estimation models.

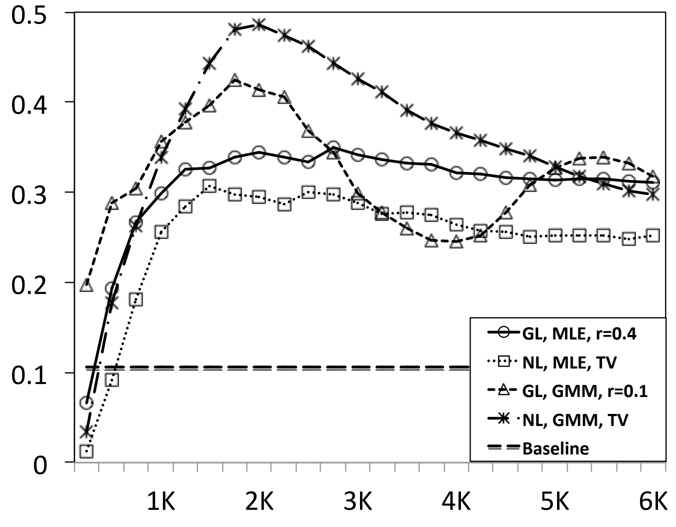
Model	Estimation	Measure	Factor	#word	ACC	AED
(1)	GMM	NL	sim_{TV}	2K	0.486	583.2
(2)	MLE	GL	$r = 0.5$	2K	0.449	611.6
(3)	MLE	GL	$r = 0.1$	2.75K	0.323	827.8

results for Models (1) and (2) are shown Fig. 2.7.

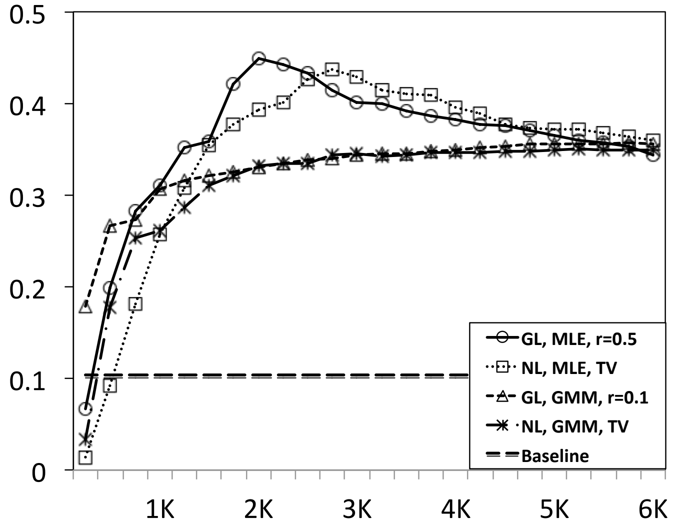
2.4.6 Smoothing vs. Feature Selection

The technique to simultaneously increase the probability of unseen terms (that cause the sparseness problem) and decrease that of seen terms is referred to as *smoothing*. While successfully applied in many IR problems, in the context of location prediction problem from Twitter data, it has been reported that smoothing has very little effect in improving the accuracy [14, 6]. On the other hand, as reported in [6], *feature selection* seems to be very effective in solving the location prediction problem. That is, instead of using the entire corpus, [6] proposed to use a selective set of "local words" only. Through the experiments, we validated that the feature selection idea via local words is indeed effective. For instance, Fig. 2.7 shows that our best results usually occur when around 2,000 local words (identified by either NL or GL methods), instead of 26,998 original terms, are used in predicting locations. Having a reduced feature set is beneficial, especially in terms of speed. For instance, with Model (1) estimated by MLE, using 50, 250, 2,000, and 26,999 local words, it took 27, 32, 50, and 11,531 seconds respectively to finish the prediction task. In general, if one can get comparable results in ACC and AED, solutions with a smaller feature set (i.e., less number of local words) are always preferred. As such, in this section, we report our exploration to reduce the number of local words used in the estimation even further.

Figs. 2.5–2.7 all indicate that both ACC and AED (in all settings) improve in proportion to the size of local words up to 2K–3K range, but deviate afterwards. In particular, note that those high-ranked words within top-300 (according to NL or GL measures) may be good local words but somehow have limited impact toward overall ACC and AED. For instance, using GMM as the estimation model, GL yields the following within the top-10 local words: {`windstaerke`, `prazim`, `cnen`}. Upon inspection, however, these words turn out to be Twitter user IDs. These words got high local word scores



(a) Model (1)



(b) Model (2)

Figure 2.7. Settings for two models to achieve the best ACC.

(i.e., GL) probably because their IDs were used in re-tweets or mentioned by users with a strong spatial pattern. Despite their high local word scores, however, their usage in the entire corpus is relatively low, limiting their overall impact. Similarly, using MLE as the estimation mode, NL found the followings at high ranks: {`je`, `und`, `kt`}. These words are Dutch (thus not filtered in preprocessing) and heavily used in only a few US towns[2] of Dutch descendants, thus exhibiting a strong locality. However, again, their overall

[2]Nederland (Texas), Rotterdam (New York), and Holland (Michigan)

Table 2.6. Prediction with reduced # of local words by frequency.
(a) Model (1), GMM, NL

		Number of local words used				
		50	*100*	*150*	*200*	*250*
ACC	Top 2K	0.433	0.447	0.466	0.476	**0.499**
	Top 3K	0.446	0.449	0.444	0.445	0.446
AED	Top 2K	603.2	599.6	582.9	565.7	531.1
	Top 3K	**509.3**	567.7	558.9	539.9	536.5

(b) Model (1), MLE, GL

		Number of local words used				
		50	*100*	*150*	*200*	*250*
ACC	Top 2K	0.354	0.382	0.396	0.419	**0.420**
	Top 3K	0.397	0.400	0.399	0.403	0.416
AED	Top 2K	771.7	761.0	760.3	730.6	**719.8**
	Top 3K	806.2	835.1	857.5	845.9	822.3

(c) Model (3), MLE, GL

		Number of local words used				
		50	*100*	*150*	*200*	*250*
ACC	Top 2K	0.2227	0.276	0.315	0.336	0.343
	Top 3K	0.301	0.366	0.385	0.401	**0.408**
AED	Top 2K	743.9	663.3	618.5	577.7	570.3
	Top 3K	620.7	565.6	535.1	510.8	**503.3**

impact is very limited due to the rarity outside those towns. From these observations, therefore, we believe that both localness as well as frequency information of words must be considered in ranking local words.

Informally, $score(w) = \lambda \frac{localness(w)}{\Delta_l} + (1 - \lambda)\frac{frequency(w)}{\Delta_f}$, where Δ_l and Δ_f are normalization constants for $localness(w)$ and $frequency(w)$ functions, and λ controls the relative importance between localness and frequency of w. The localness of w can be calculated by either NL or GL, while frequency of w can be done using IR methods such as relative frequency or TF-IDF. For simplicity, in this experiments, we implemented the $score()$ function in two-steps: (1) we first select base 2,000 or 3,000 local words by NL or GL method; and (2) next, we re-sort those local words based on their frequencies. Table 2.6 shows the results of ACC and AED using only a small number (i.e., 50–

Table 2.7. Top-30 Frequency-resorted local words (GMM, NL).

la	nyc	hiring	dallas	francisco
obama	fashion	atlanta	houston	denver
san	diego	sf	austin	est
chicago	los	seattle	hollywood	yankees
york	boston	washington	angeles	bears
ny	miami	dc	fl	orlando

250) of top-ranked local words after re-sorted based on both localness and frequency information of words. Note that using only 50–250 local words, we are able to achieve comparable ACC and AED to the best cases of Table 2.5 that use 2,000–3,000 local words. The improvement is the most noticeable for Model (1). The results show the quality of the location prediction task may rely on a small set of frequently-used local words.

Table 2.7 shows top-30 local words with GMM, when re-sorted by frequency, from 3,000 NL-selected words. Note that most of these words are *toponyms*, i.e., names of geographic locations, such as `nyc`, `dallas`, and `fl`. Others include the names of people, organizations or events that show a strong local pattern with frequent usage, such as `obama`, `fashion`, or `bears`. Therefore, it appears that toponyms are important in predicting the locations of Tweeter users. Interestingly, a previous study in [14] showed that toponyms from image tags were helpful, though *not* significantly, in predicting the location of the images. Table 2.8 shows the results using city names with the highest population in U.S. gazetteer as the “only” features for predicting locations (without using other local words). Note that performances are all improved with all three models, but are not good as those in Table 2.6. Therefore, we conclude that using toponyms in general improve the prediction of locations, but *not* all toponyms are equally important. Therefore, it is *important to find critical local words or toponyms using our proposed NL or GL selection methods*. It further justifies that such a selection needs to be made from the evidences in tweet contents and user location, rather based on semantic meanings or types of words (as [6] did).

Table 2.8. Prediction with only toponyms.

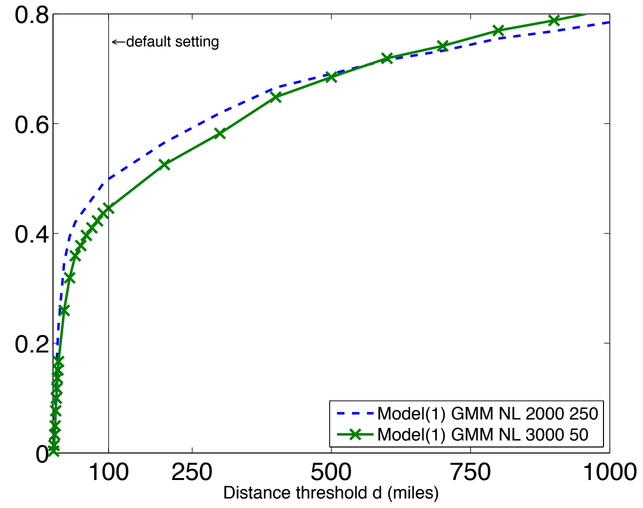
		Number of toponyms used		
		50	*200*	*400*
ACC	Model (1)	0.246	0.203	0.115
	Model (2)	0.306	0.291	0.099
	Model (3)	0.255	**0.347**	0.330
AED	Model (1)	1202.7	1402.7	1719.7
	Model (2)	741.9	953.5	1777.4
	Model (3)	668.2	512.4	**510.1**

2.4.7 Discussion on Parameter Settings

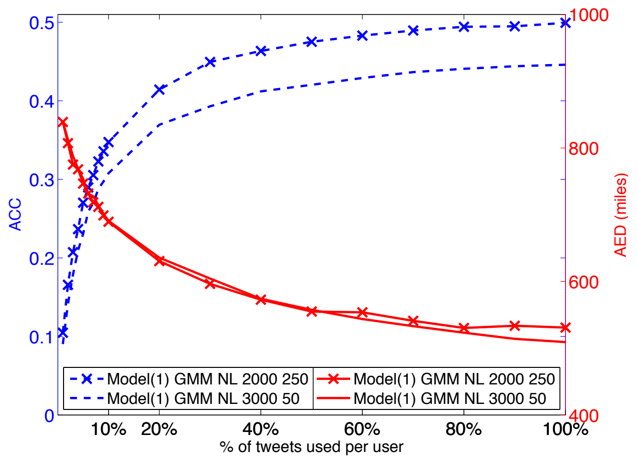
First, same as the setting in literature, we used $d = 100$ (miles) in computing ACC–i.e., if the distance between the estimated city and ground truth city is less than 100 miles, we consider the estimation to be correct. Fig. 2.8(a) shows that ACC as a function of d using the best configuration (Model (1), GMM, NL) with 50 and 250 local words, respectively. Second, the test set that we used in experiments consists of a set of active users with around 1K tweets, same setting as [6] for comparison. Since not all Twitter users have that many tweets, we also experiment using different portion of tweet messages per user. That is, per each user in the test set, we randomly select from 1% to 90% of tweet messages to predict locations. The average results from 10 runs are shown in Fig. 2.8(b). While we achieve ACC (shown in left Y-axis) of 0.104 using 1% (10 tweets) per user, it rapidly improves to 0.2 using 3% (30 tweets), and 0.3 using 7% (70 tweets). Asymmetrically, AED (shown in right Y-axis) decreases as tweets increases.

2.5 Conclusion

In this chapter, we proposed a novel approach to estimate the spatial word usage probability with Gaussian Mixture Models. We also proposed unsupervised measurements to rank the local words which effectively remove the noises that are harmful to the prediction. We show that our approach can, using less than 250 local words, achieve a comparable or better performance to the state-of-the-art that uses 3,183 local words selected by the supervised classification based on 11,004 hand-labeled ground truth.



(a) Varying d



(b) Varying $|T_u|$

Figure 2.8. Change of ACC and AED as a function of d and $|T_u|$.

Chapter 3

Improving Recommendation Using User-Generated Relationships

3.1 Introduction

Thanks to the innovation of social networks, various types of user activities are recorded, such as following, liked, comment and so on. These kinds of behaviors can be viewed as users expressing their preferences, and are very common in many social network websites such as Facebook, Google+, and so on. To study them and make recommendation for items that user would like have been one of the most important tasks for modern recommender system researchers. Recently, Netflix Prize showed promising results for movie recommendation. However the explicit feedback studied in Netflix, such as 1-5 stars in movie ratings, is very limited and often expensive to acquire. In contrast, the implicit feedback, such as inferred preferences, are more prevalent but also less informative, presenting both opportunities and challenges. In addition, the studies in movie recommendation aim to improve the overall performances, while in item recommendation the performance of top ranked results is more important.

Another challenges in studying user's preference is that the data, despite prevalent, still suffers from the data spareness problem due to long tail phenomenon. One possible remedy to reduce the effect is to enrich the data by taking related information into consideration. However, the recent studies on item recommendation often focus on a single task, and overlook the interactions between different types of activities of a user. Studies in rating prediction in movie recommendation has shown that the correlation in the ratings of an user and the ratings of her friends (i.e. the homophily effect[21]) can be used to improve the rating prediction. We are interested in finding if similar correlations can be found in different types of like, and if they can be used to improve item recommendation. On Flickr, for example, a user expresses her preference on an image by tagging it as her favorite, on an user by following her, or on a community(group) by subscribe to it (called group). Often, we can find some correlation between different types of 'likes'. For example, a user who shared a animal image shows she has some interest on animals, and is likely to tag a animal image from others as her favorite, or join a group of animal photos. In addition, a group on Flickr also collects a set of images related to its theme. Thus we can extend the like relationship to different types of subjects. That is, a group likes an image. It is also possible that the likes between different type of subjects might also be correlated. That is, a user likes a group might also like the images the group likes, or in other direction, a user might like the images collected by his liked group.

To further study the relationships between different subjects involved in a social network, we present a view of a system. We view a social network like Flickr a system of various types of entities such as users, groups, and images, each has a like relationship with each other, or itself. We categorize the like relationships by the involved types of entities into two kinds: inter-relationship if the liking is between different types of entity (e.g. user to image), and self-relationship if the liking is between same type of entity (e.g. user to user). In our Flickr example, as shown in Fig 3.1, a user has an inter-relationship with an image by 'upload' and 'tag', and with group by 'join' and 'discuss', while a group has an inter-relationship with an image by 'collect'. An entity might also have a relationship with others of the same type, such as a user following another user (friendship), or an image similar another image. Each of the relationships in the system presents an unique recommendation task, for example, recommending users to user or recommending images to user. While the current studies for item recommendation, tend to tackle each individually, we are interested in improving the tasks collectively. The presented view of a social network system is rather general, not specific to Flickr. For example, Facebook and Google+ both have community which has its own actions similar to a user, such as posting an message or a photo. Therefore, our view also applies to other system.

In this chapter, we first present an analysis in finding correlation between differ-

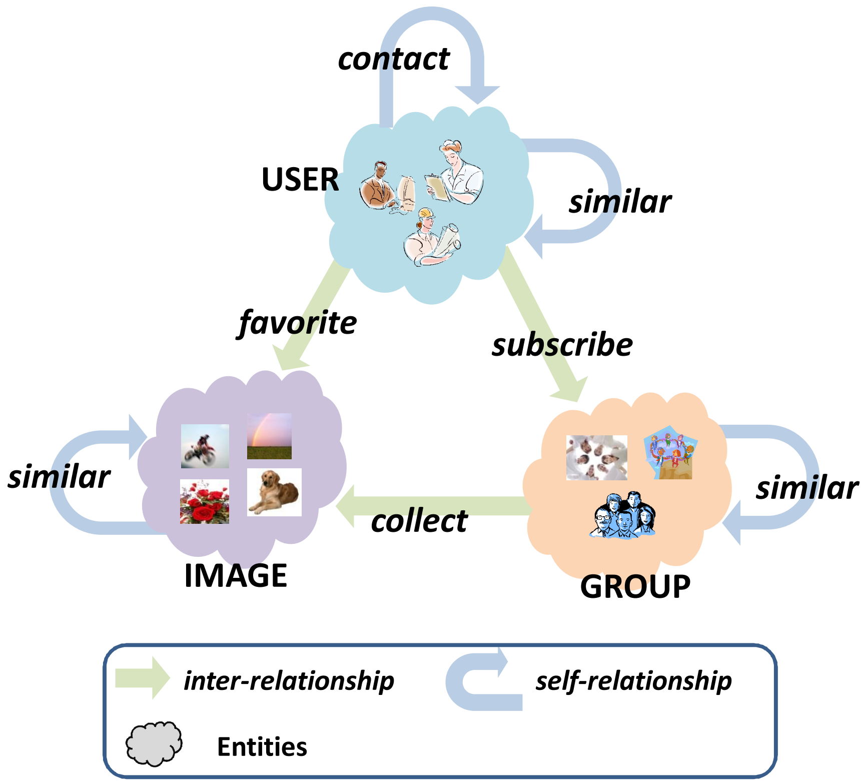


Figure 3.1. Multiple Entities with Relationships on Flickr

ent types of like for users. We then propose different algorithms to take advantage of the correlation. We show that different tasks might enrich each other and improve the performance of each.

3.1.1 Problem Definition

3.1.1.1 Notations

In this section, following our new view on a social network system, we discuss the notation and present our tasks to be studied in our example system Flikcr. The followings guide the notations used in this paper.

- Rating Matrix R. A superscript might be used to indicate the corresponding rating matrix. R^{UG}, R^{UI} and R^{GI} is the rating matrix for user-group, user-image and group-images, respectively.

- Weighting Matrix W. The matrix used to develop Weighted Regularized Matrix

Factorization. Similar to R, a superscript might be indicates the specific matrix.

- Latent factor matrix. U, G and I is the low rank latent matrix for user, group and image, respectively. Each is of the size of the number of subjects/items times the rank K. When the symbols are used in superscript with a matrix, it indicates the specific matrix corresponds to them. For example, S^U is the *self-relationship* matrix for user. With they are with subscript, it indicate the latent vector with respect to the subject/item. For example, I_k is the latent vector for image k.

- *self-relationship* matrix S. An element S_{ij} in a *self-relationship* matrix S indicates a existing friendship or similarity connection between i and j.

- Vector. We use the subscript $i\cdot$ or $\cdot j$ with a matrix symbol to indicate a row vector and a column vector, respectively. For example, $R^{UG}_{i\cdot}$ and $W^{UG}_{\cdot j}$ indicates subject i's rating vector and item j's weighting vector, respectively.

- Diagonal matrix. We use $\widetilde{}$ above a vector to indicate a diagonal matrix whose diagonal is the vector. For example, $\widetilde{W^{UI}_{\cdot k}}$ is a diagonal matrix whose diagonal is the vector $W^{UI}_{\cdot k}$.

- Scalar value. A matrix with two index symbols in the subscript indicates a specific value matrix. For example, $R^{GI}_{j,k}$ is the rating of group j on image k.

- Index variable. h, i, j, k are used to index a matrix or vector.

- Rank K. K is the dimensionality of the latent factor.

- Size. The number of users, groups and images is N_U,N_G and N_I respectively.

- Regularization parameter. λ denotes the parameter for controlling the level of regularization.

- $\|\cdot\|_F$ denotes the Frobenius norm.

- $\mathbb{I}$ denotes identity matrix.

- $\approx$ denotes approximation.

- $\mathrm{nz}()$ is a function which defines the number of non-zero elements in a matrix or vector.

3.1.1.2 Recommendation Tasks

We are interested in the following four tasks: The tasks we study in this paper are defined as follows.

- Group Recommendation for User: recommending groups that a user is most likely to join. $R^{UG} \in \{0,1\}^{N_U \times N_G}$ represents the user-group matrix.
- Image Recommendation for User: recommending images that a user is most likely to tag as favorite. $R^{UI} \in \{0,1\}^{N_U \times N_I}$ represents the user-image matrix.
- Image Recommendation for Group: recommending images that a group is most likely to collect. $R^{GI} \in \{0,1\}^{N_G \times N_I}$ represents the user-image matrix.
- Friend Recommendation for User: recommending images that a user is most likely to make friend. $S^{U} \in \{0,1\}^{N_U \times N_U}$ represents the user-user matrix.

To abbreviate, we refer them in the form of ”entity-entity”, such as ”user-group”. Our broader view exceed the traditional user-item view, therefore we use ”subject” instead of ”user”.

The values in the rating matrices are inferred from the recorded actions. We take the All Missing As Negative(AMAN) notion [22], where unobserved value are treated as negative class. For example, if user u joins group g then $R^{UG}_{ug} = 1$. Similarly, if user u tags image i or group g collects image i, then $R^{UI}_{ui} = 1$ or $R^{GI}_{gi} = 1$. Otherwise, the values are zero.

3.2 Related Work

Collaborative filtering (CF) has been studied extensively in the recent years. In general, the studies [23] fall in three categories: memory-based, model-based, and hybrid. Memory-based approach typically involves aggregate the ratings of most similar users or items as the prediction for a subject. In model based approach, the recommender system tries to identify the latent factors that explain the existing ratings. In particular, the low-rank matrix factorization methods have been shown very successful for predicting movie ratings in the Netflix Prize competition [24]. Compared to the explicit feedback where users give definite ratings within a fixed range, such as [0,5] or good,bad, recommendations for implicit feedback, such as playing times for song or subscriptions of an interested group, are less studied. The challenges are different for the two types of data: a rating in a fixed range is predicated with explicit feedback, while a small set of items which the user is most likely interested is recommended with implicit feedback. Different evaluation measures are used accordingly: RMSE or MAE are typically used for explicit data, and ranked score such as NDCG, MAP, MRR, and Top-k scores are often used for implicit data. Thus the recommendation with implicit feedback is closely connected to studies for Top-k recommendation.

Pan et al. [25] proposed a specific problem setup One Class Collective Filtering (OCCF) where the data only positive examples can be observed. Weighted Regulated Matrix Factorization (WRMF) proposed by Hu et al. [25] and Pan et al. [22] which uses different confidence level for the observed and unobserved data. More recently, several matrix factorization techniques are proposed to improve Top-k recommendation by designing different optimization criteria. Bayesian Personalized Ranking (BPR) proposed by [26] aims to maximize the pair-wise differences between relevant and irrelevant items, which results to optimize the Area Under the Curve (AUC) measure. However, it is argued [27] that AUC might not be good measure for the recommendation task. Collaborative Less-is More Filtering (CLiMF), Shi et al. [27], overcomes the difficulty to directly optimize MRR measure due to its non-smoothed function by approximate it with smoothed function of MRR to allow it to be optimized computationally. On the similar idea, Shi et al. [28] proposed TFMAP optimize MAP measure. Although MRR and MAP well reflect ranked results, the optimization requires high numerical stability in computation which might present challenges for practical use.

Incorporating side information to improve recommendation has been proved useful in recent studies. The studies usually take conventional user-item view, and seek to take more information on users or items in consideration. The most often utilized side information including item description, user profile [29], trust network, and social network [30]. User profile and item description are usually used similar to memory-based approach where similarities between users or items are calculated in order to find neighbors for them. Under the assumption that similar users (or items) might have similar behaviors, the recommendation might get improved by referring to their most similar neighbors. Similarly, trust network or social network might benefit the recommendation if users have similar taste to their sccial friends or users gained their trust. In particular, SocialMF [31] includes the trustee's latent factors into the user factor during the optimization, which leads the implicit information propagation along the trusting links. SoRec [32] which co-factorize user-item matrix along with the user's trusted network. These studies focus on *self-relationships* as illustrated in Fig 3.1, and the *inter-relationships* are investigated to our best knowledge.

3.3 Proposed Methods

3.3.1 Analysis on Flickr Dataset

3.3.1.1 Network Structure Analysis

MediaEval provides public benchmarking datasets to evaluating algorithms for multimedia access and retrieval. We use the randomly sampled users in MediaEval 2011 dataset as seed users and crawl their related information. We then select users which has at least 10 contacts, 10 subscribed groups, and 10 favorite images to form our set of user U. We also select groups G and images I which have least 10 connections to each other into our dataset. From the collected dataset, we build the following like matrices: user-group (UG), user-image (UI), group-image(GI), and user-user (friendship) (Su).

The homophily phenomenon indicates that friends in a social network tend to have similar interests. Several recent studies [32, 31] have shown improved performances by taking items liked by friends interest into consideration. To understand the difference between friendship network and other types of likes, we first compare the network structures of the like matrices, and then evaluate the correlations between them. Table 3.1 describes the size and the density of the matrices. Despite items with low interactions are filtered out, the matrices are still 99% sparse or lower, indicating that the recommendation tasks have high difficulty. Figure 3.2 demonstrates the degree distribution of the studied matrices. The degree of user in UI and Su are very similar. The numbers of users is similar in UG and Su when the node degree is small, while all other distributions approximately follow the power law pattern.

Each like matrix consists of two types of entity, and forms a bipartite network. We

Table 3.1. Description of Flickr Dataset.

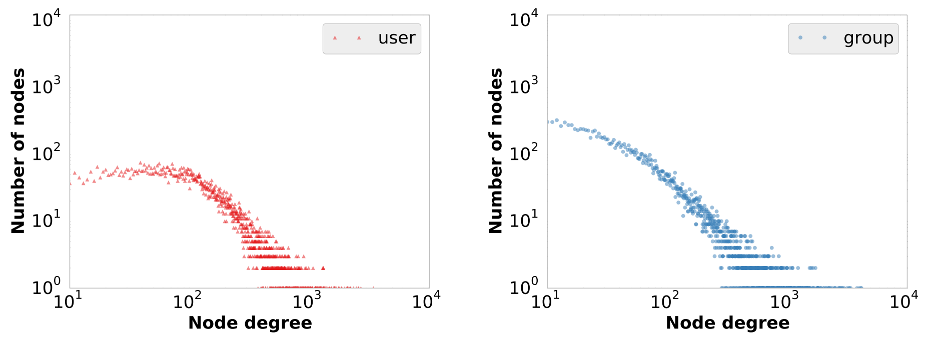
	Dimensionality	Density
user-group	10592×15625	1.04%
user-image	10592×62008	0.16%
group-image	15625×62008	0.26%
user-user	10592×10592	0.63%

perform bipartite network analysis on them (shown in Table 3.2), and compare them to the friendship network (shown in Table 3.3) with different network characteristics, such as average degree, clustering coefficient, and centrality. Clustering coefficient measures the tendency of vertices clustering in a graph. For sparse matrix, in general, networks higher average number of degree provide more information, while higher clustering coefficient indicates vertices are more likely to group with others. The closeness centrality measures the shortest path distances of a vertex to others, which provide another perspective on how the network is structured. Higher centrality shows the vertices are closer to each other on average.

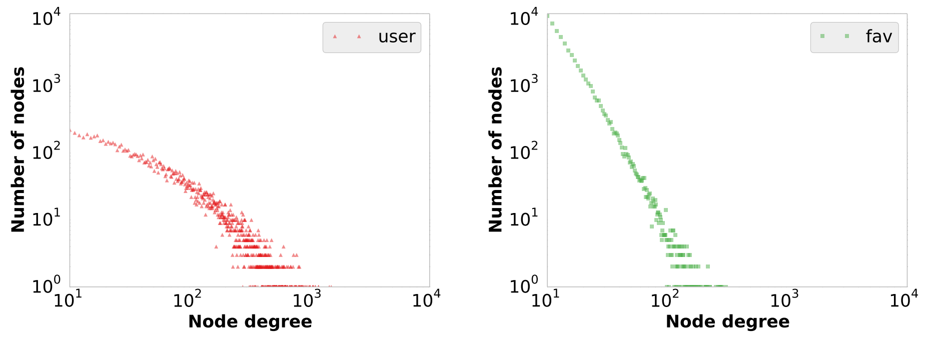
While the user social network are users' direct preferences on other users, the user-group and user-image bipartite network can be considered as indirect connections between users, through common liked groups or images. The mean degree of friends in Table 3.3) considers both in and out degrees. By considering only out degree, the number of users a user follows, it is 67.13, larger than than the number of liked group, 13, but fewer than the number of liked images, 100.4. The closeness centrality also exhibits a reverse pattern, partially due to the higher density, that users are closer in user-group than social network, and in turn, than user-image. A network with high clustering effect tends to have better capability to categorize the nodes. In our analysis, the social network shows strongest clustering compared to the other two user networks. Among the later two types of connection, the groups is a better medium to connect users then does the images. In sum, the user friendship network has distinctively higher clustering coefficient, but is similar to other types of network for the other measures.

3.3.1.2 Correlation Analysis

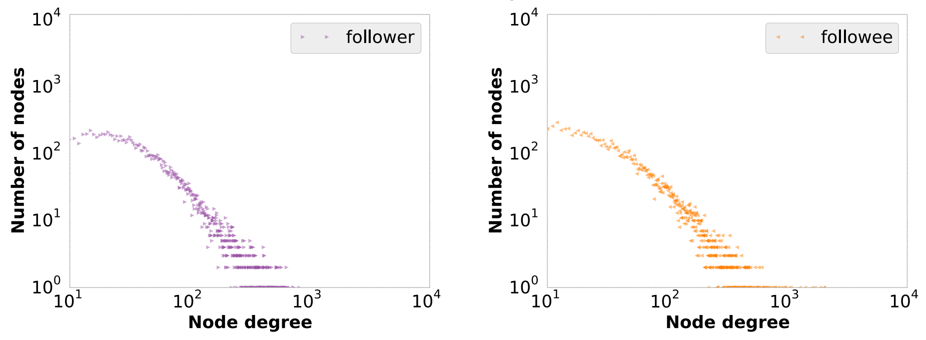
In this section, we discuss how to validate the correlation between different types of likes by asking the question: Can a user liked groups be recovered by in the groups collecting



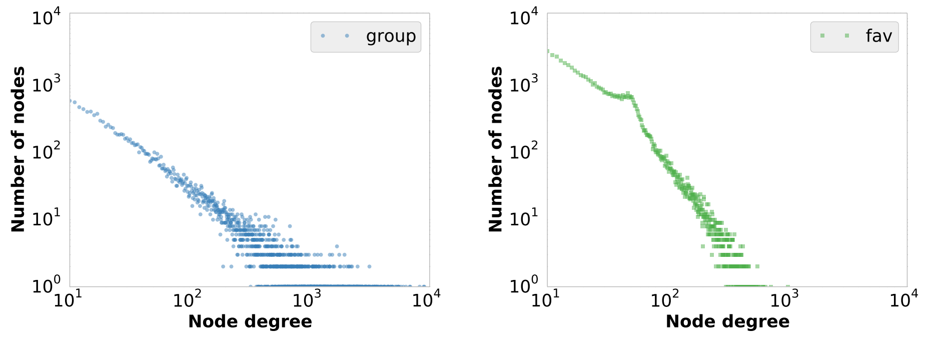
(a) user-group UG



(b) user-image UI



(c) user-user Su



(d) group-image GI

Figure 3.2. Degree Distribution in Flickr Dataset.

Table 3.2. Bipartite Like Network Analysis on Flickr Dataset.

(a) **user-group**

Mean	Degree	Clustering coefficient	Closeness centrality
user	13.0	0.024	0.51
group	162.7	0.018	0.53

(b) **user-image**

Mean	Degree	Clustering coefficient	Closeness centrality
user	100.4	0.008	0.33
image	17.2	0.035	0.49

(c) **group-image**

Mean	Degree	Clustering coefficient	Closeness centrality
group	163.6	0.011	0.36
image	41.2	0.029	0.54

Table 3.3. Friendship (user-user) Network Analysis on Flickr Dataset.

Mean	Degree	Clustering coefficient	Closeness centrality
user-user	134.3	0.12	0.41

her liked images? Here we define the preferences directly expressed by user as first order relationships such as liked groups or liked images. The groups collecting liked images by a user is inferred indirectly from a first order relationship, hence a second order relationship. If the two first-order relationships are closely related, we might be able to infer one relationship through a second order relationship. For example, a user expresses her interests with the liked images and liked groups. If the liked items exhibit consistent interests from a user, we might find overlaps between her liked imaged and images collected by her liked group. From the perspective of information retrieval (IR), we view the items in the first order relationship as the relevant items and try to retrieve them through a second order relationship. Therefore, metrics from IR such as precision and recall can be used to measure the quantity of overlaps for a user, defined as

$$\text{recall}(u) = \frac{|FO(u) \cap SO(u)|}{|FO(u)|}$$

Table 3.4. Results of Second Order Likes Retrieval.

	precision	recall	F1
liked images, retrieved by random groups	0.0016	0.2534	0.0032
liked images, retrieved by liked groups	0.0031	**0.6908**	0.0061
liked images, retrieved by liked users	0.0057	0.3980	**0.0113**
liked groups retrieved by random images	0.0236	0.2713	0.0434
liked groups retrieved by liked images	0.0496	0.4220	**0.0888**
liked groups, retrieved by liked users	0.0289	**0.8336**	0.0559

$$\text{precision}(u) = \frac{|FO(u) \cap SO(u)|}{|SO(u)|}$$

. The $FO(u)$ is the set of items in a first order relationship for user u. For example, $\{i|i$ is an image liked by $u\}$. The $SO(u)$ is the set of items in a second order relationship. For example, $\{i|g$ likes i, and g is a group liked by $u\}$. Since friendship networks works well in improving recommendation task, we use it as a reference to compare the other types of like relationship. We also compare the result of random selection of groups for each user. The result are from groups that are uniformly randomly selected for each user according her number of liked groups, and are averaged from 20 times repetitions. In the results, as demonstrated in Table 3.4, the friendships (liked users) can retrieve around 40% of liked images (recall) for a user on averages, while the liked groups can retrieve around 70%, both better than 25% of random selection. In general, the higher recall shows more overlaps, thus more correlation. The recall is also affected by degree of second order relationship. For example, the recall will be 1 if a user or a group likes all the images. The case is rare, however, as we learned in the degree distribution. To take the number of items involved into account, we also compared a more balanced measure F1 score (2*(precision*recall)/(recision+recall0), which showed friendships (0.011) actually have better quality than liked group (0.006). In another test, we retrieve liked groups for users by their friends and their liked images, and found the latter has better correlation than the former. Overall, the studied three types of like relationship are correlated to others, but the quality may vary.

In this chapter, we study several algorithm that exploit the connections between different like relationships. We first discuss the baseline method and present results from one of the stat-of-the-arts algorithms for item recommendation, which makes effective recommendation for an individual task. We then propose algorithms under a co-factorization framework to collectively optimize multiple tasks.

3.3.2 Weighted Influences (WINF)

We have shown that items in a first order relationship can be recalled by a second order relationship in Sec 3.3.1.2. We further extend it by ranking the results by the frequency of a returned item. Intuitively, if an item is liked more times in a second order relationship, it should be more relevant to a user. For example, a user likes both group animal and group dog. Both groups like image dog1, while group dog also likes image dog2. We can infer that the user likes the image dog1 with a frequency 2, and dog2 with a frequency 1. Thus we return [dog1, dog2] as an ordered recommendations. Formally, to recommend an ordered list of groups $\mathrm{rec}(u)$ to user u,

$$\begin{aligned}\mathrm{rec}(u) =& \mathrm{argsort}_{\geq}(R^{UG}_{u,\cdot} \times R^{GI}) \\ =& \mathrm{argsort}_{\geq}\big((R^{UG}_{u,\cdot} R^{GI}_{\cdot,1}, \cdots, R^{UG}_{u,\cdot} R^{GI}_{\cdot,N_I})\big)\end{aligned}$$

, where $\times$ denotes matrix multiplication and $\mathrm{argsort}_{\geq}$ is a function returning the indexes that reorders the array in decreasing order. $R^{UG}_{u,\cdot} R^{GI}_{\cdot,i}$ is the frequency for image i. We can apply the same principle to other tasks as well. The advantage of this method is computational efficient and model free. However, its effectiveness less comparable as we will show in the later section. We use this algorithm as a baseline for performance comparison.

3.3.3 Same Characteristics Principle

To exploit various relationships in a given social network, we propose a multiple matrices co-factorization framework based the ‘same characteristics principle’. Balanced with scalability and accuracy, low rank matrix factorization (MF) technique based collaborative filtering is proven to be one of the most effective methods in recommender system. The underlying idea for matrix factorization is to find a small number of latent factors for subjects and items that best explains the ratings. Typically, a rating matrix is decomposed into a latent user matrix and a latent item matrix, which represent their inherent characteristics. Here we propose 'same characteristics principle' which essentially assumes all entities has a consistent set of characteristics across the system, even for different tasks. Therefore, for a subject or an item, its latent factors should remain fixed for all tasks. We argue the assumption is more reasonable than using different factors for different tasks. Under the principle, a user should have consistent interests for her selection on the groups and images, which also fits better for general people in the real world. The principle implies that the latent space is all shared for different entities. That is, the underlying representation for each dimension in the the latent space is also fixed. For example, the all entities are factored into a two-dimensional space ¡black&white, color¿ and all tasks are explained with these two dimensions. Although matrix factorization in general does not have such interpretable factors, we use it for the sake of explanation. By sharing a common latent space across the system, the data sparseness could be alleviated when a subject with few ratings in one task might be helped by another task. Furthermore, regularization is also strengthened since a common set of latent factors now subject to multiple task matrices, thus reducing the chance of over-training.

Another advantages of the same characteristics principle is that it naturally leads to a unified co-factorization framework which is capable to accommodate rich information and different relationships. In general, the framework can be implemented with any MF technique. Among the state-of-the-art MFs for item recommendation, *WRMF* is balanced with good performance and efficiency. Thus, we implement our framework with *WRMF*. We introduce *WRMF* in the next section, then discuss how to incorporate different type of relationship gradually in the following.

3.3.4 *WRMF* : Weighted Regularized Matrix Factorization

Weighted Regularized Matrix Factorization[22, 25] is proposed to make item recommendation for One Class Collaborative Filtering. It naturally extends the matrix factorization techniques for the explicit rating recommendation task by incorporating a weighting matrix to indicate the confidences. A existing rating gives more confidence, thus heavier weight is given ($1+\alpha$) in the corresponding W, as opposed to the 1 for empty rating. Despite the simplicity of design, its performance has been shown as one of tops among other state-of-the-arts algorithms for the task, such as BPR[26], CLiMF[27], TFMAP[28], and SLIM[33]. The simplicity leads to better efficiency in computation, thus more favorable as opposed to others. Therefore, we choose *WRMF* as the base matrix factorization technique to implement our co-factorization framework. Another advantage of *WRMF* is that the model training can be carried out by Alternative-Least-Squares (ALS) optimization which makes parallel implementation possible. In general, to approximate matrix R with low rank factor matrices U and V, *WRMF* minimizes the object function:

$$\mathcal{L}_{\textit{WRMF}} = \sum_{ij} W_{ij}(R_{ij} - U_i V_j^T)^2 + \lambda(\|U\|_F^2 + \|V\|_F^2)$$

, where

$$W_{ij} = \begin{cases} 1 & \text{if } R_{ij} = 0 \\ 1 + \alpha & \text{if } R_{ij} = 1 \end{cases}$$

3.3.5 *WRS* : *WRMF* with *self-relationship*

Homophily phenomenon [21] states people with social ties tend to have similar characteristics or interests. Studies that exploits this phenomenon report promising results for explicit ratings, such as SoRec[32] and SocialMF[31]. From from the of view of items, a similar connection can be formed by the similarity between two from the associate information, such as item descriptions. Studies[34, 35] also report improvements by taking item's side information into consideration. In our extended view, we consider both of them a case of *self-relationship* , since both of the participating entities are of them same type. Here we express a *self-relationship* in the from of matrix by S^U, S^G and S^I, representing a *self-relationship* network for user, group and image, respectively.

Friendship is a natural *self-relationship* between users, which might exist in different forms. On Flickr, the social relationship is uni-directional and in one of the categories: 'following', 'friends' and 'family', depending on the tie between the two. The latter two also imply the 'following'. In our study, we use 'following' to represent the social relationship. By the social relationship, we define S^U as

$$S_{ij}^U = \begin{cases} 1 & \text{if user } i \text{ follows } j \\ 0 & \text{otherwise} \end{cases}$$

Content based filtering has been shown effective by utilizing textual information associated with subjects or items, such as user profile[29], search history[36]. On Flickr, each group has a group name and description edited by its administrators to explain itself. With such textual information, we compute the tf-idf vector representation and derive a similarity accordingly between two. Let T_i and T_j denotes the tf-idf vector for group i and group j and index h refer to a vocabulary, the similarity score is defined as:

$$\mathrm{sim}_{textual}(i,j) = \cos(T_i, T_j) = \frac{\sum\limits_h T_{ih}T_{jh}}{\sqrt{\sum\limits_h {T_{ih}}^2}\sqrt{\sum\limits_h {T_{jh}}^2}}$$

To construct the similarity network, we can consider threshold or top-k approach. The former selects a pre-defined threshold β and make a connection if similarity score is larger than β. That is,

$$S^G_{ij} = \begin{cases} 1 & \text{if } \mathrm{sim}(i,j) \geq \beta \\ 0 & \text{if else} \end{cases}$$

. For the top-K approach, the score of k-th item is selected as β for each subject.

When such textual information is limited, we can resort to memory-based method which exploits existing ratings to compute similarity between subject. As one of the well studied collaborative filtering technique, it has been shown effective in many commercial systems. The similarity derivation is similar to the textual similarity. For group i and group j, their rating similarity with respect to the user-image is

$$\mathrm{sim}_{GI}(i,j) = \cos(R^{GI}_{i\cdot}, R^{GI}_{j\cdot}) = \frac{\sum\limits_h R^{GI}_{ih}R^{GI}_{jh}}{\sqrt{\sum\limits_h R^{GI2}_{ih}}\sqrt{\sum\limits_h R^{GI2}_{jh}}}$$

. Rating similarity with respect to the user-group sim_{GU} can also be calculated in the same fashion. The final rating similarity can be determined by weighted average with a weight γ as

$$\mathrm{sim}_{rating}(i,j) = \gamma\,\mathrm{sim}_{GI}(i,j) + (1-\gamma)\mathrm{sim}_{GU}(i,j)$$

, and following above steps to construct rating similarity network. Equivalently, for images, we can compute textual similarities with the tags specified by its owner, or

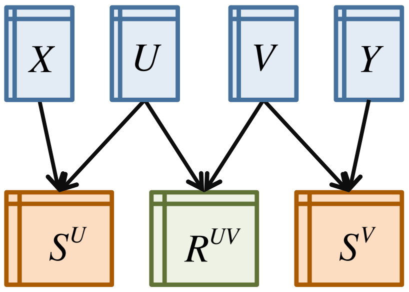


Figure 3.3. Co-factorization in *WRS*.

rating similarities with existing ratings. One or a combination of the similarities is then used to construct similarity network S^I.

As most studies discussed above evaluate *self-relationship* with explicit feedback by its overall performances, the reports for its effectiveness on item recommendation with implicit feedback are still limited. Therefore we proposed *WRS*, an extension with *WRMF* to incorporate *self-relationship* such as the friendship network or similarity network from side information. For a recommendation task with observed rating matrix R^{UV}, *self-relationship* matrices S^U and S^V, *WRS* finds latent matrices U, V, X and Y such that $R^{UV} \approx UV^T$, $S^U \approx UX^T$ and $S^V \approx VY^T$. The relationships of latent matrices and rating matrices is demonstrated in Fig 3.3. *WRS* minimizes the the objective function:

$$\begin{aligned}
&\mathcal{L}_{\mathit{WRS}}(R^{UV}, S^U, S^V, U, V, X, Y) \\
&\quad = \sum_{i,j} W^{UV}_{ij}(R^{UV}_{ij} - U_i V_j^T)^2 \\
&\quad + \lambda_U \sum_{i,j} W^U_{ij}(S^U_{ij} - U_i X_j^T)^2 \\
&\quad + \lambda_V \sum_{i,j} W^V_{ij}(S^V_{ij} - V_i Y_j^T)^2 \\
&\quad + \lambda(\|U\|_F^2 + \|V\|_F^2 + \|X\|_F^2 + \|Y\|_F^2)
\end{aligned}$$

Here, UV represents a general subject-item task. $\|U\|_F^2$, $\|V\|_F^2$, $\|X\|_F^2$, and $\|Y\|_F^2$

are regularized terms to control the model complexities with λ. With same characteristics principle, a common latent space matrix U for subjects is shared for both rating matrix R^{UV} and *self-relationship* matrix S^{U}, thus connecting the two tasks the matrices represent. The same go with the items as well. All tasks are weighted regularized by their corresponding weighting matrices, e.g. W^{UV} to R^{UV}. λ_U and λ_V are used to control to impact from the *self-relationship*. If it was set to 0, the information from the *self-relationship* matrix will be nullifies. As it increases, its impact gets larger in the cost function, which essentially also lowers the impact from other matrices.

The objection function *WRI* can be minimized with the Alternative-Least-Squares. We first take derivative of $\mathcal{L}_{\textit{WRS}}$ for each latent vector and set it to zero, that is, set $\frac{\partial \mathcal{L}_{\textit{WRS}}}{\partial U_i} = 0$, $\frac{\partial \mathcal{L}_{\textit{WRS}}}{\partial V_i} = 0$, $\frac{\partial \mathcal{L}_{\textit{WRS}}}{\partial X_i} = 0$ and $\frac{\partial \mathcal{L}_{\textit{WRS}}}{\partial Y_i} = 0$. Then we can derive the updating rules as the following.

$$U_i = \left(R^{UV}_{i\cdot}\widetilde{W^{UV}_{i\cdot}}V + \lambda_U S^{U}_{i\cdot}\widetilde{W^{U}_{i\cdot}}X\right)\left(V^T\widetilde{W^{UV}_{i\cdot}}V + \lambda_U X^T\widetilde{W^{U}_{i\cdot\cdot}}X + \lambda\mathbb{I}\right)^{-1} \tag{3.1a}$$

$$V_i = \left(R^{UV}_{\cdot i}\widetilde{W^{UV}_{\cdot i}}U + \lambda_V S^{V}_{i\cdot}\widetilde{W^{V}_{i\cdot}}Y\right)\left(U^T\widetilde{W^{UV}_{\cdot i}}U + \lambda_V Y^T\widetilde{W^{V}_{i\cdot}}Y + \lambda\mathbb{I}\right)^{-1} \tag{3.1b}$$

$$X_i = \lambda_U S^{U}_{\cdot i}\widetilde{W^{U}_{\cdot i}}U\left(\lambda_U U^T\widetilde{W^{U}_{\cdot i}}U + \lambda\mathbb{I}\right)^{-1} \tag{3.1c}$$

$$Y_i = \lambda_V S^{V}_{\cdot i}\widetilde{W^{V}_{\cdot i}}V\left(\lambda_V V^T\widetilde{W^{V}_{\cdot i}}V + \lambda\mathbb{I}\right)^{-1} \tag{3.1d}$$

The detail step for train a model is described in Alg 1. Latent matrices are initialized by assigning random small values in the beginning. In each iteration in the optimization, only one factor matrix makes update at a given time and depends only on others whose values are fixed. For example, updating U only requires V and X, and both are fixed until U is done. Therefore, different subjects U_i can be updated in parallel.

As suggested in [25], by rewriting $(V^T\widetilde{W^{UV}_{i\cdot}}V)$ to $(V^TV - V^T(\widetilde{W^{UV}_{i\cdot}} - \mathbb{I})V)$, the computation time can be shortened, since now $(\widetilde{W^{UV}_{i\cdot}} - \mathbb{I})$ only has $\mathrm{nz}(R^{UV}_{i\cdot})$ non-zero elements and V^TV can be computed beforehand and outside of iterations. $\mathrm{nz}(R)$ indicates the number of non-zero elements in R. In a iteration of updating Ui, shown in Eq (3.1a), the time complexity for the first term $(R^{UV}_{i\cdot}\widetilde{W^{UV}_{i\cdot}}V + \lambda_U S^{U}_{i\cdot}\widetilde{W^{U}_{i\cdot}}X)$ is $O(\mathrm{nz}(R^{UV}_{i\cdot})K + \mathrm{nz}(W^{U}_{i\cdot})K)$, since only non-zero terms needs to be involved in the calculation. K is the dimensionality of a latent factor. The second term ,$(V^T\widetilde{W^{UV}_{i\cdot}}V + \lambda_U X^T\widetilde{W^{U}_{i\cdot\cdot}}X + \lambda\mathbb{I})^{-1}$ has time complexity $O(\mathrm{nz}(R^{UV}_{i\cdot})K^2 + \mathrm{nz}(S^{U}_{i\cdot})K^2 + K^3)$ with the above speed-up. The matrix inversion operation is assumed to take $O(K^3)$. The overall time complexity for updating U is $O(\mathrm{nz}(R^{UV})K^2 + \mathrm{nz}(S^U)K^2 + N_U K^3)$ by aggregating all iterations, where N_U is the number of subjects in U. Since K is typically small, the computation largely depends on number of observed ratings and the number of subject. Since updating of each iteration is fully parallel-able, the overall time can speed up around n times with n-way parallelism. Similar analysis applies for Eq (3.1c) and Eq (3.1d).

Algorithm 1: Learning Algorithm for *WRS*

```
input : Ratings matrices $R^{UV}$, $S^U$, $S^V$, maximum number of iteration maxiter,
        parameters $\lambda$,$\lambda_U$, $\lambda_V$
output: Latent matrices $U, V, X, Y$

Initialize $U, V, X, Y$ with random small values;
for $t \leftarrow 1$ to maxiter do
    parallel for $i \leftarrow 1$ to $N_U$ do
        Update $U_i$ according to Eq (3.1a);
    end
    parallel for $i \leftarrow 1$ to $N_V$ do
        Update $V_i$ according to Eq (3.1b);
    end
    parallel for $i \leftarrow 1$ to $N_X$ do
        Update $X_i$ according to Eq (3.1c);
    end
    parallel for $i \leftarrow 1$ to $N_Y$ do
        Update $Y_i$ according to Eq (3.1d);
    end
end
```

3.3.6 *WRI* : *WRMF* with *inter-relationship*

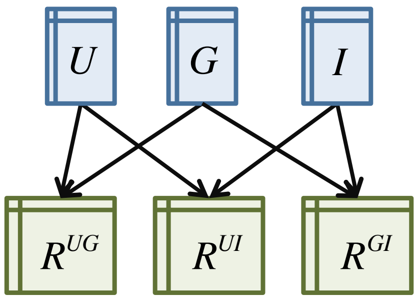


Figure 3.4. Co-factorization in *WRI* .

As we learn in Sec 3.3.1.2, there are also correlations exists between different tasks, similar to correlation between users in friendship network. Unlike *self-relationship* , the connects are made between different types of entity through a common subject. For example, the same user expresses interests in groups and images. To exploit this correlation, we propose *WRI* which solves three studied recommendation tasks globally and collectively. With the same principle, multiple matrices are factorized with a common set of latent factors for each entity. With the rating matrices R^{UG}, R^{UI} and R^{GI}, *WRS* finds latent matrices U, G and I, representing user, group, image, respectively, such that $R^{UG} \approx UG^T$, $R^{UI} \approx UI^T$ and $R^{GI} \approx GI^T$. The relationship between rating matrices and latent matrices are demonstrated in Fig 3.4. Specifically, *WRI* optimizes the objective function:

$$\begin{aligned}
&\mathcal{L}_{\textit{WRI}}(R^{UG}, R^{UI}, R^{GI}, U, G, I) \\
&\quad = \sum_{i,j} W_{ij}^{UG}(R_{ij}^{UG} - U_i G_j^T)^2 \\
&\quad + \lambda_{UI} \sum_{i,k} W_{ik}^{UI}(R_{ik}^{UI} - U_i I_k^T)^2 \\
&\quad + \lambda_{GI} \sum_{j,k} W_{jk}^{GI}(R_{jk}^{GI} - G_j I_k^T)^2 \\
&\quad + \lambda(\|U\|^2 + \|G\|^2 + \|I\|^2)
\end{aligned}$$

λ_{UI} and λ_{GI} are used to adjust relatively weights between the three task. $\|U\|^2, \|G\|^2$ and $\|I\|^2$ are regularization terms, controled by λ. Similar to *WRS* , the objection function *WRI* can be minimized with ALS, which results in the following updating rules.

$$\begin{aligned}
U_i &= \left(R_{i\cdot}^{UG}\widetilde{W_{i\cdot}^{UG}}G + \lambda_{UI}R_{i\cdot}^{UI}\widetilde{W_{i\cdot}^{UI}}I\right)\left(G^T\widetilde{W_{i\cdot}^{UG}}G + \lambda_{UI}I^T\widetilde{W_{i\cdot}^{UI}}I + \lambda\mathbb{I}\right)^{-1} \\
G_j &= \left(R_{\cdot j}^{UG}\widetilde{W_{\cdot j}^{UG}}U + \lambda_{GI}R_{j\cdot}^{GI}\widetilde{W_{j\cdot}^{GI}}I\right)\left(U^T\widetilde{W_{\cdot j}^{UG}}U + \lambda_{GI}I^T\widetilde{W_{j\cdot}^{GI}}I + \lambda\mathbb{I}\right)^{-1} \\
I_k &= \left(\lambda_{GI}R_{\cdot k}^{GI}\widetilde{W_{\cdot k}^{GI}}G + \lambda_{UI}R_{\cdot k}^{UI}\widetilde{W_{\cdot k}^{UI}}U\right)\left(\lambda_{GI}G^T\widetilde{W_{\cdot k}^{GI}}G + \lambda_{UI}U^T\widetilde{W_{\cdot k}^{UI}}U + \lambda\mathbb{I}\right)^{-1}
\end{aligned}$$

The optimization steps is detailed in Alg 2. Following the analysis in Sec3.3.5, the overall time complexity for updating U is $O(\mathrm{nz}(R^{UG})K^2 + \mathrm{nz}(R^{UI})K^2 + N_U K^3)$.

Algorithm 2: Learning Algorithm for *WRI*

```
input : Ratings matrices $R^{UI}$, $R^{UG}$, $R^{GI}$, maximum number of iteration **maxiter**,
        parameter $\lambda$, $\lambda_{UI}$, $\lambda_{GI}$
output: Latent matrices $U, G, I$

Initialize $G$,$I$ with random small values;
for $t \leftarrow 1$ to maxiter do
    parallel for $i \leftarrow 1$ to $M$ do
        Update $U_i$ according to Eq (??);
    end
    parallel for $j \leftarrow 1$ to $N$ do
        Update $G_j$ according to Eq (??);
    end
    parallel for $k \leftarrow 1$ to $L$ do
        Update $I_k$ according to Eq (??);
    end
end
```

3.3.7 *WRCO* : Multiple Relationships Co-Factorization

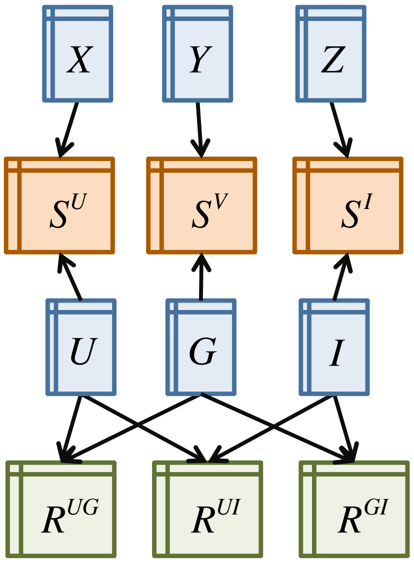


Figure 3.5. Co-factorization in *WRCO* .

We propose *WRCO* to exploit both *self-relationship* and *inter-relationship* to connect multiple tasks and side information under a unified framework. With rating matrices R^{UG}, R^{UI} and R^{GI}, *self-relationship* matrices S^U, S^G and S^I, *WRS* finds latent matrices U, G, I, X, Y and Z, such that $R^{UG} \approx UG^T$, $R^{UI} \approx UI^T$, $R^{GI} \approx GI^T$,

$S^U \approx UX^T$, $S^V \approx VY^T$ and , $S^I \approx IZ^T$. The relationship between rating matrices and latent matrices are demonstrated in Fig 3.5. Combining both *WRS* and *WRI* , the object function for *WRCO* is defined as

$$
\begin{aligned}
&\mathcal{L}_{WRCO}(R^{UG}, R^{UI}, R^{GI}, S^U, S^G, S^I, U, G, I, X, Y, Z) \\
&\quad = \sum_{i,j} W_{ij}^{UG}(R_{ij}^{UG} - U_i G_j^T)^2 \\
&\quad + \lambda_{UI} \sum_{i,k} W_{ik}^{UI}(R_{ik}^{UI} - U_i I_k^T)^2 \\
&\quad + \lambda_{GI} \sum_{j,k} W_{jk}^{GI}(R_{jk}^{GI} - G_j I_k^T)^2 + \lambda_U \sum_{i,h} W_{ih}^{U}(S_{ih}^{U} - U_i X_h^T)^2 \\
&\quad + \lambda_G \sum_{j,h} W_{jh}^{G}(S_{jh}^{G} - G_j Y_h^T)^2 + \lambda_I \sum_{k,h} W_{kh}^{I}(S_{kh}^{I} - I_k Z_h^T)^2 \\
&\quad + \lambda(\|U\|_F^2 + \|G\|_F^2 + \|I\|_F^2 + \|X\|_F^2 + \|Y\|_F^2 + \|Z\|_F^2)
\end{aligned}
$$

Similar to *WRS* and *WRI* , we optimize *WRCO* with ALS with the following updating rules.

$$
\begin{aligned}
U_i = &\left(R_{i\cdot}^{UG}\widetilde{W_{i\cdot}^{UG}}G + \lambda_{UI}R_{i\cdot}^{UI}\widetilde{W_{i\cdot}^{UI}}I + \lambda_U S_{i\cdot}^{U}\widetilde{W_{i\cdot}^{U}}X\right) \\
&\left(G^T\widetilde{W_{i\cdot}^{UG}}G + \lambda_{UI}I^T\widetilde{W_{i\cdot}^{UI}}I + \lambda_U X^T\widetilde{W_{i\cdot.}^{U}}X + \lambda\mathbb{I}\right)^{-1} \qquad (3.2a) \\
G_j = &\left(R_{\cdot j}^{UG}\widetilde{W_{\cdot j}^{UG}}U + \lambda_{GI}R_{j\cdot}^{GI}\widetilde{W_{j\cdot}^{GI}}I + \lambda_G S_{j\cdot}^{G}\widetilde{W_{j\cdot}^{G}}Y\right) \\
&\left(U^T\widetilde{W_{\cdot j}^{UG}}U + \lambda_{GI}I^T\widetilde{W_{j\cdot}^{GI}}I + \lambda_G Y^T\widetilde{W_{j\cdot}^{G}}Y + \lambda\mathbb{I}\right)^{-1} \qquad (3.2b) \\
I_k = &\left(\lambda_{GI}R_{\cdot k}^{GI}\widetilde{W_{\cdot k}^{GI}}G + \lambda_{UI}R_{\cdot k}^{UI}\widetilde{W_{\cdot k}^{UI}}U + \lambda_I S_{k\cdot}^{I}\widetilde{W_{k\cdot}^{I}}Z\right) \\
&\left(\lambda_{GI}G^T\widetilde{W_{\cdot k}^{GI}}G + \lambda_{UI}U^T\widetilde{W_{\cdot k}^{UI}}U + \lambda_I Z^T\widetilde{W_{k\cdot}^{I}}Z + \lambda\mathbb{I}\right)^{-1} \qquad (3.2c) \\
X_i = &\ \lambda_U S_{\cdot i}^{U}\widetilde{W_{\cdot i}^{U}}U\left(\lambda_U U^T\widetilde{W_{\cdot i}^{U}}U + \lambda\mathbb{I}\right)^{-1} \qquad (3.2d) \\
Y_j = &\ \lambda_G S_{\cdot j}^{G}\widetilde{W_{\cdot j}^{G}}G\left(\lambda_G G^T\widetilde{W_{\cdot j}^{G}}G + \lambda\mathbb{I}\right)^{-1} \qquad (3.2e) \\
Z_k = &\ \lambda_I S_{\cdot k}^{I}\widetilde{W_{\cdot k}^{I}}I\left(\lambda_V I^T\widetilde{W_{\cdot k}^{V}}I + \lambda\mathbb{I}\right)^{-1} \qquad (3.2f)
\end{aligned}
$$

The update procedure is detail in Alg 3. Following the previous analysis, The time complexity for for updating U in *WRCO* is then $O(\mathrm{nz}(R^{UG})K^2 + \mathrm{nz}(R^{UI})K^2 + +\mathrm{nz}(S^U)K^2 + N_U K^3)$.

Algorithm 3: Learning Algorithm for *WRCO*

```
input  : Ratings matrices $R^{UI}, R^{UG}, R^{GI}, S^U, S^G, S^I$, maximum number of iteration
         maxiter, parameter $\lambda$, $\lambda_{UI}$, $\lambda_{GI}$, $\lambda_U$,$\lambda_G$,$\lambda_I$
output: Latent matrices $U, G, I, X, Y, Z$

Initialize $U$,$G$,$I$,$X$,$Y$,$Z$ with random small values;
for $t \leftarrow 1$ to maxiter do
    parallel for $i \leftarrow 1$ to $N_U$ do
        Update $U_i$ according to Eq (3.2a);
    end
    parallel for $j \leftarrow 1$ to $N_G$ do
        Update $G_j$ according to Eq (3.2b);
    end
    parallel for $k \leftarrow 1$ to $N_I$ do
        Update $I_k$ according to Eq (3.2c);
    end
    parallel for $i \leftarrow 1$ to $N_U$ do
        Update $X_i$ according to Eq (3.2d);
    end
    parallel for $j \leftarrow 1$ to $N_G$ do
        Update $Y_j$ according to Eq (3.2e);
    end
    parallel for $k \leftarrow 1$ to $N_I$ do
        Update $Z_k$ according to Eq (3.2f);
    end
end
```

3.4 Experimental Validation

3.4.1 Experimental Setup

We use two types of data split to examine our ideas. One is the commonly used cross validation. The ratings in the studied matrices are randomly shuffled and are split into 6 folds. We run each experiment 5 times. In each iteration, we use 4 folds as training set, 1 fold as development set to select best parameters, and the remaining 1 fold as validation set. The average results from validating set are reported. In the prepared data, the average degree can as high as more than 100 in our dataset depending the matrices, as shown in the Table 3.2. In the real worlds, the number might be too ideal since the dataset is prepared by filtering out subject with very low number of ratings in order to study the relationship between different likes. Therefore, a given K split strategy is also employed, where only K ratings from each subject is studied. We set K to 20, which is acceptable even for users with infrequent usage. In this split, we use 20 randomly selected ratings are used for each subject to train a model. The remaining ratings are split into 5 folds where 1 fold is used to tune parameters and 4 folds are used for validation. We ran the process 5 times, and report the average results.

In the ideas with matrix factorization, the factor matrices are randomly initialized which may slightly affect the final results. To stabilize the comparison, we generated a set of factor matrices beforehand and use them in the same type of experiment.

We studied our results from different perspectives. RMSE and MAE are commonly used in the studies for explicit feedback to measure the overall performance of the prediction. In contrast, for item recommendation, top ranked of the predicted items is more important, since users are often only interested in top items suggested items from recommender system. Therefore, we use MRR and top-k metrics to measure the ranking performance. MRR reflects the rank of the first relevant items, while top-k metrics measure the ranking up to k returns. To reflect the overall ranking performance, we used NDCG which is a average score weighted by ranks. An item is relevant for an subject if there exists such a rating in the testing set. Ratings in the training set are simply ignored. The followings give a formal definition of the used metrics.

- Mean Reciprocal Rank(MRR). The average reciprocal ranks for the first relevant item, defined as

$$\frac{1}{n(S)} \sum_{i=1}^{n(S)} \frac{1}{\textit{rank of first relevant item for subject } i}$$

- Normal Discounted Cumulative Gain (NDCG). NDCG works on the all the return predictions and measures the overall ranking quality of relevant predictions, defined as NDCG=DCG/IDCG, where

$$DCG = \sum_{p=1}^{n(I)} \frac{2^{rel(p)}}{log(p+1)}$$

, where p is the position of each ranked prediction. $rel(p) = 1$ if the item at p is relevant, and $rel(p) = 0$, otherwise. IDCG is the maximum possible DCG for a set of predictions.

- Precision at k (Prec@k). Prec@k measures the performance of top k predictions in term of precision defined as

$$Prec@k = \frac{\textit{number of relevant items in top k prediction}}{k}$$

- Recall at k (Recall@k). Recall@k measures the performance of top k predictions in term of recall defined as

$$Recall@k = \frac{\textit{number of relevant items in top k prediction}}{\textit{number of all relevant items}}$$

To conclude over performance of an idea, we present averages from all the tasks. Since the number of subjects are different, two types of average are used: *mean*, the simple arithmetic mean from all tasks, and *weighted mean* (w. mean) average weighted by the number of subjects in a task.

3.4.2 Baseline

We first compare the difficulty of different tasks with different number of factors K in factorization with regard to different metrics. For *CV*, as demonstrated in Table 3.6a and Table 3.6b, the respective difficulties for the the studied tasks correspond to the densities of the rating matrices. Performance with regard to individual metric increase as K increases. The user-group has best performance in all metrics, and followed by user-user, group-image and user-image. Results from *Given 20*, as demonstrated in Table 3.7a and Table 3.7b, has some differences compared to the results of *CV*. The user-user tops user-group in term of the Recall@20, while the difficulties follows the same order as in *CV* for all other metrics despite all matrices now have similar densities.

The performance might not be improved the increase of the number of factors. For example, user-group has the top performance at $K = 5$, and gradually declines as K increases. The user-user tops at $K = 20$ for MRR and Prec@20.

Table 3.5 compares the performances of *WRMF* with number of factors of 5, and weighted influences (WINF) with likes and friends. *WRMF* outperforms the WINF for all task in all metrics, while WINF with friendship outperforms WINF with different likes.

Table 3.5. Comparison of *WRMF* and WINF.

(a) **user-group**

	MRR	NDCG	Recall@20	Prec@20
WRMF_nf5	0.008	0.287	0.036	0.038
WINF_likes	0.002	0.183	0.005	0.003
WINF_friends	0.003	0.192	0.010	0.007

(b) **user-image**

	MRR	NDCG	Recall@20	Prec@20
WRMF_nf5	0.007	0.356	0.070	0.066
WINF_likes	0.002	0.189	0.010	0.007
WINF_friends	0.003	0.187	0.011	0.009

(c) **group-image**

	MRR	NDCG	Recall@20	Prec@20
WRMF_nf5	0.321	0.383	0.083	0.101
WINF_likes	0.036	0.198	0.009	0.008

3.4.3 Results of *WRS*

To learn the effectiveness of *WRS*, we study several networks including friendship network, group similarity network, and image similarity network. Friendship network straightforward since the connections are expressed directly by users. The similarity network can to be inferred from different perspectives. In this experiment, we use textual similarity to construct the network by the description and title of groups and the tags and title of the images. We also need to decide the k to select the top-k most similar items to build the connections. We first test different k to learned their behaviors in the *WRS* with varifying λ. The λ is used to control the impact of the *self-relationship*.

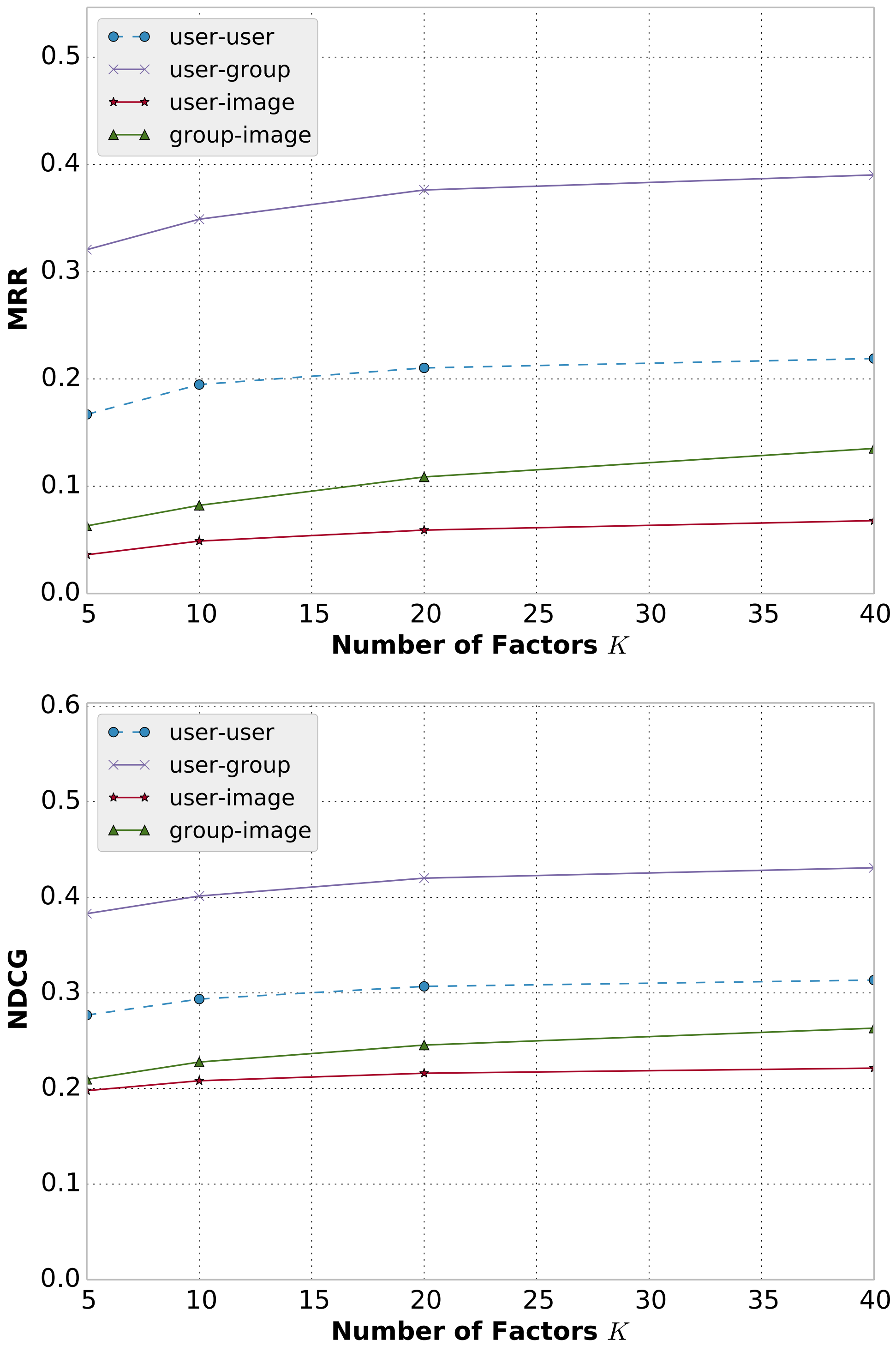
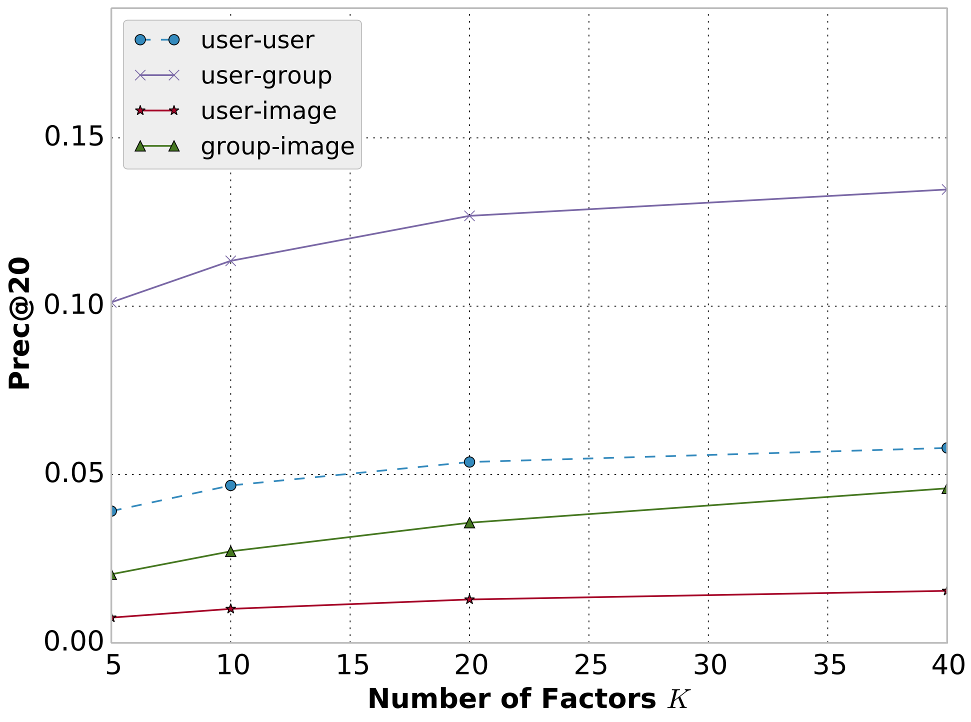


Figure 3.6a. Comparison of *CV* Tasks with *WRMF*.



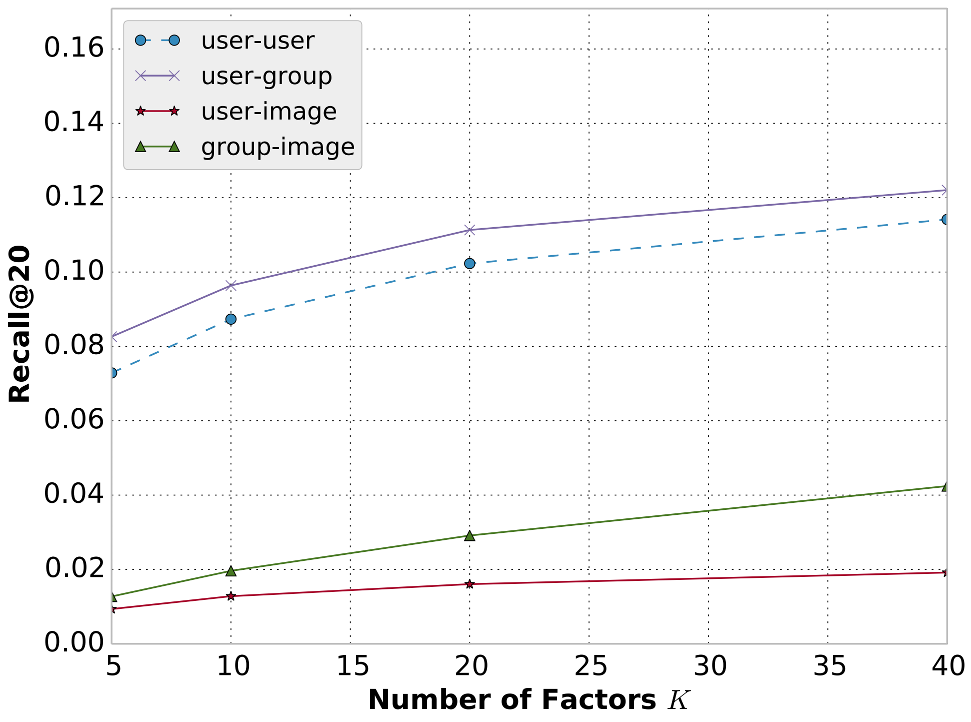
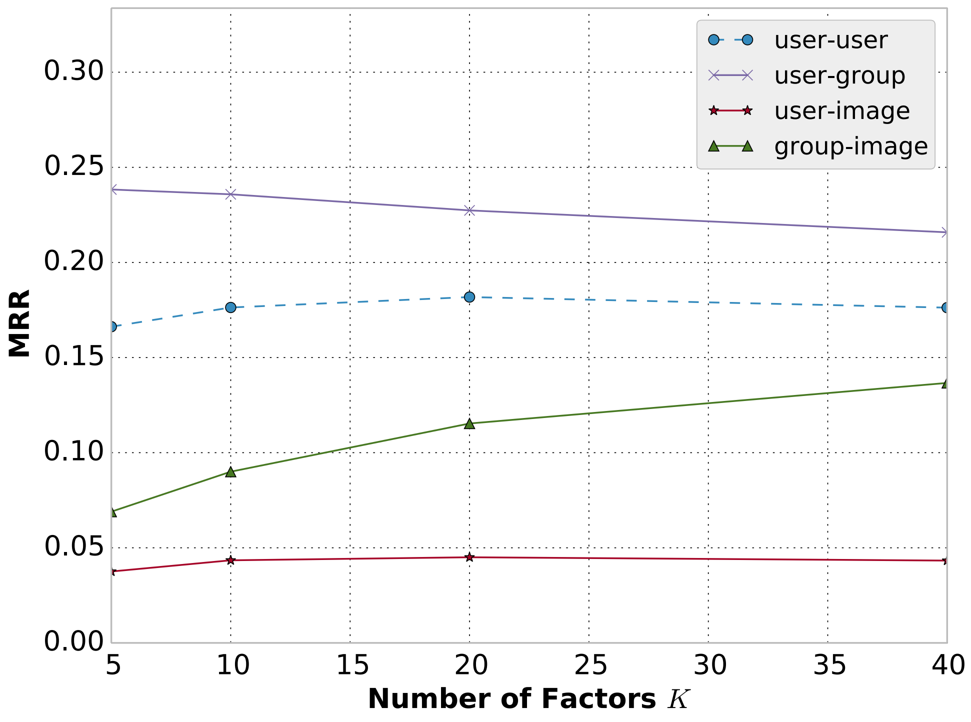


Figure 3.6b. Comparison of *CV* Tasks with *WRMF*.



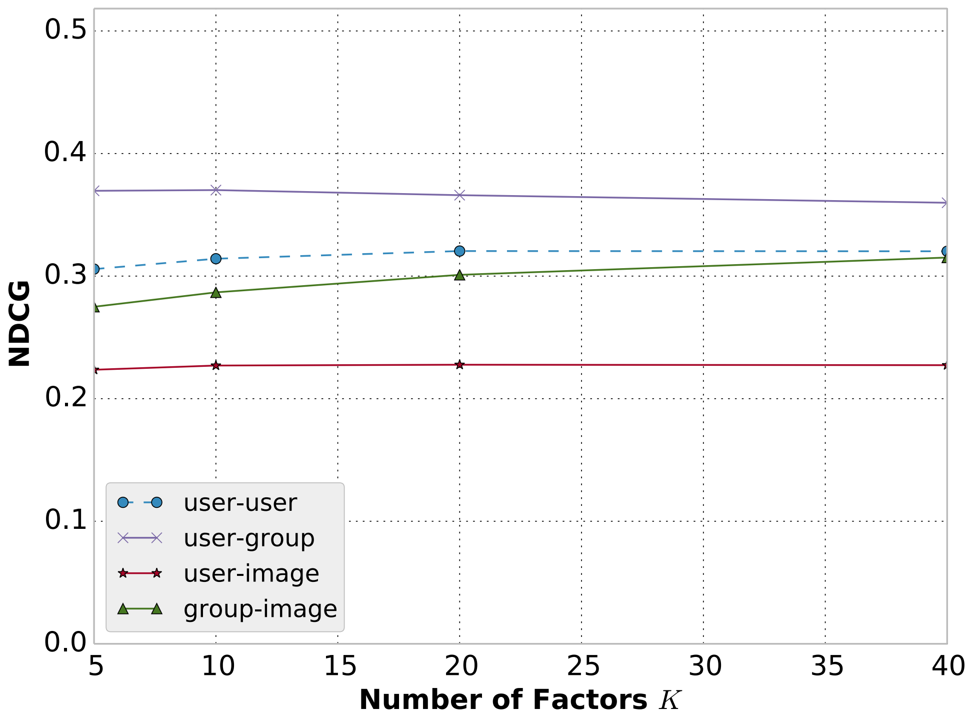


Figure 3.7a. Comparison of *Given 20* Tasks with *WRMF* .

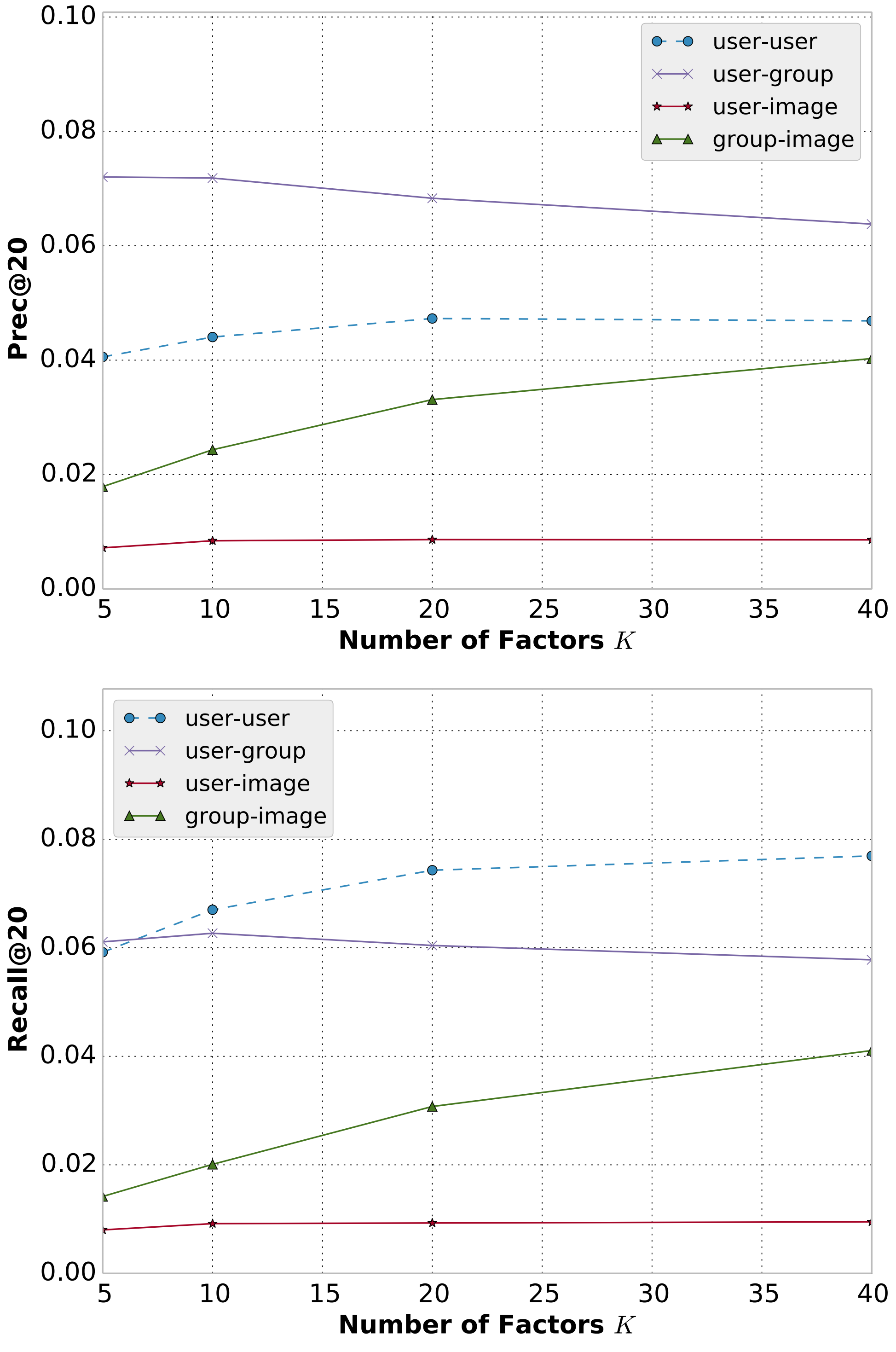
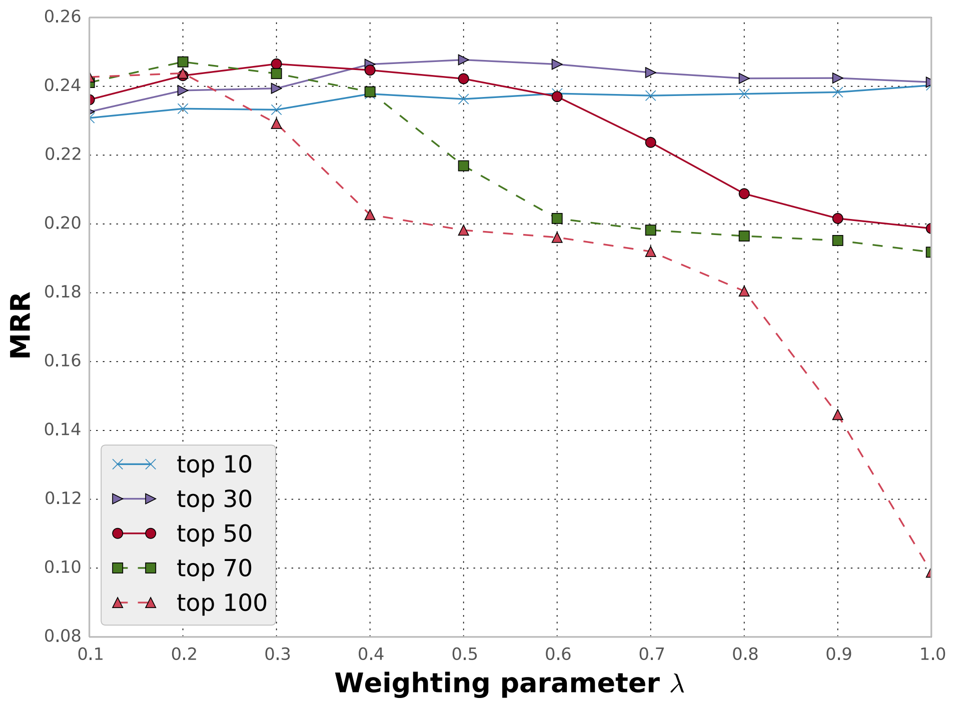
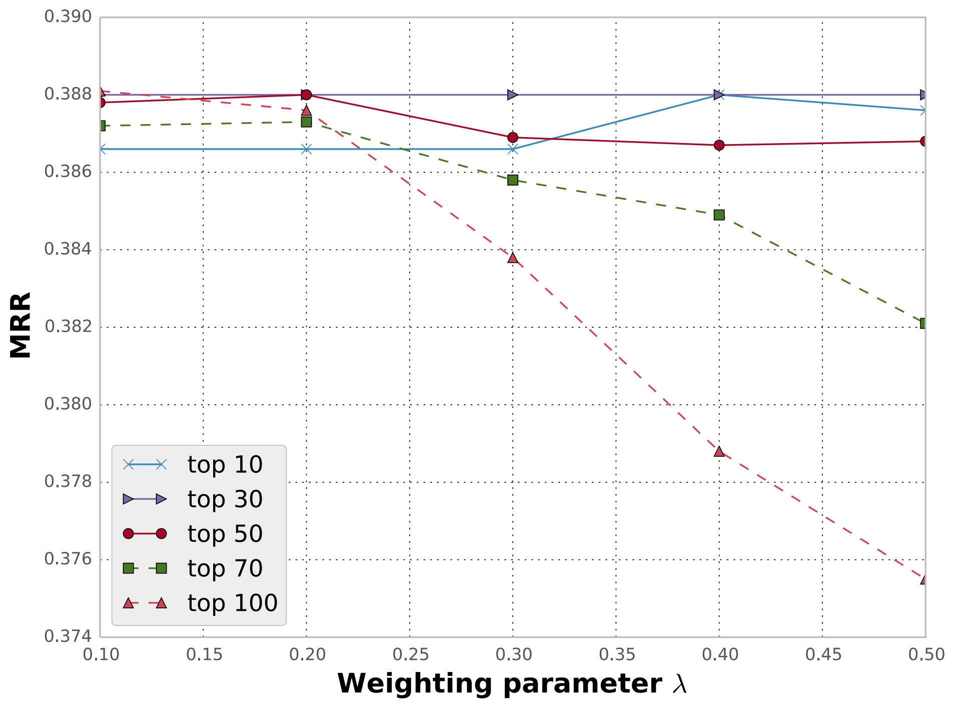


Figure 3.7b. Comparison of *Given 20* Tasks with *WRMF* .



(a) S^G, *CV*, with MRR



(b) S^G, *Given 20*, with MRR

Figure 3.8. Selecting top-k Most Similar Items in the Network.

When set to 0, *WRS* degenerates to *WRMF*. Table 3.8 shows the results of MRR of selecting different k for the user-group tasks. Both data splits shows stable results with $k = 30$. Therefore, we use it for S^G. For S^I, different k shows only marginal impact on the results. We also use $k = 30$ for S^I.

We then study impact of *self-relationship* individually on different tasks. To present an overview on the results, we report the relative improvements over *WRMF* aggregated from the studied four metrics MRR, NDCG, Prec@20, and Recall@20. For *CV*, as shown in Table 3.6, S^U improve both user-group and user-image tasks by 6% and 2% as K increases, while S^G and S^I improve user-group and user-image by 3% and 1% , respectively. The task group-image are not helped with both S^G and S^I. In general, larger K shows better improvement. For *Given 20*, as shown in Table 3.7, S^U improve both user-group and user-image tasks by 20% and 22% as K increases, while S^G and S^I improve user-group and user-image by 13% and 5%, respectively. Compared to *CV*, the improvement is more significant, indicating better effectiveness of *self-relationship* when data is few. Similar to *CV*, group-image is only marginal improved. For both data split, S^U has most improvement, while the S^I has the least improvement.

Table 3.6. Relative Improvement of *WRS* over *WRMF*, *CV*.

(a) S^U

K \Tasks	user-group	user-image
20	0.55%	0.72%
40	1.68%	2.48%
80	6.46%	2.79%

(b) S^G

K \Tasks	user-group	group-image
20	0.23%	-0.60%
40	0.62%	-0.24%
80	3.78%	0.01%

(c) S^I

K \Tasks	user-image	group-image
80	1.56%	0.51%

We also present the impact of varifying λ. Fig 3.9a shows the contribution of SU with the different λ while Fig 3.9b shows that of SG. In general, we can find a pattern

Table 3.7. Relative Improvement of *WRS* over *WRMF*, *Given 20*.

(a) S^U

K \Tasks	user-group	user-image
10	5.41%	3.43%
20	12.92%	12.24%
40	20.69%	22.81%

(b) S^G

K \Tasks	user-group	group-image
10	1.07%	1.68%
20	6.25%	0.42%
40	13.90%	0.47%

(c) S^I

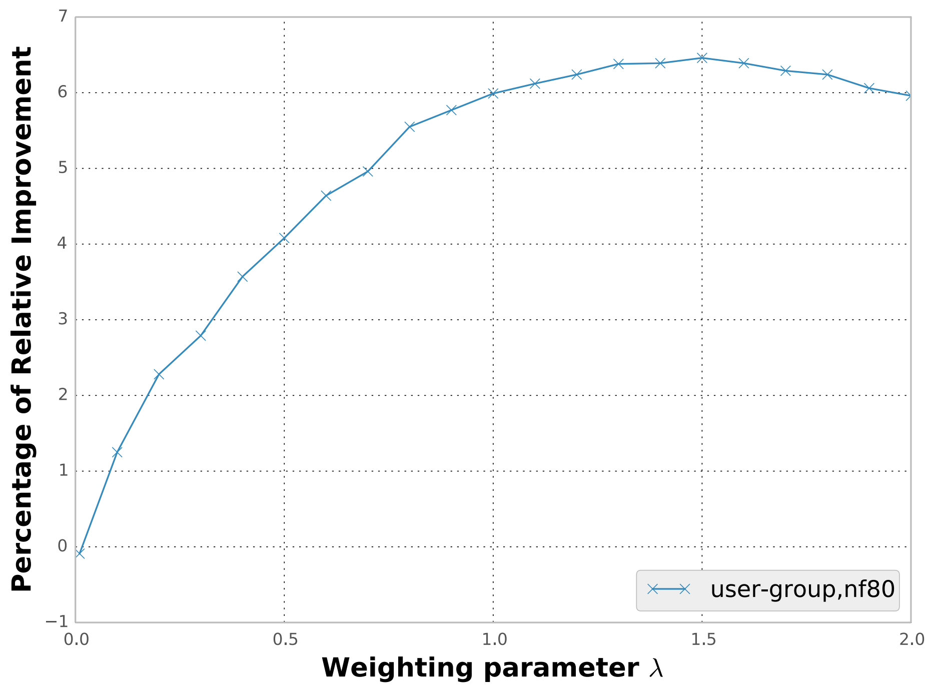
K \Tasks	user-image	group-image
40	5.70%	0.25%

that the performance improves as λ increases. After certain point, the performance declines as λ increases. The feature helps to find the optimal performance by a greedy strategy in the grid search of the best parameter.

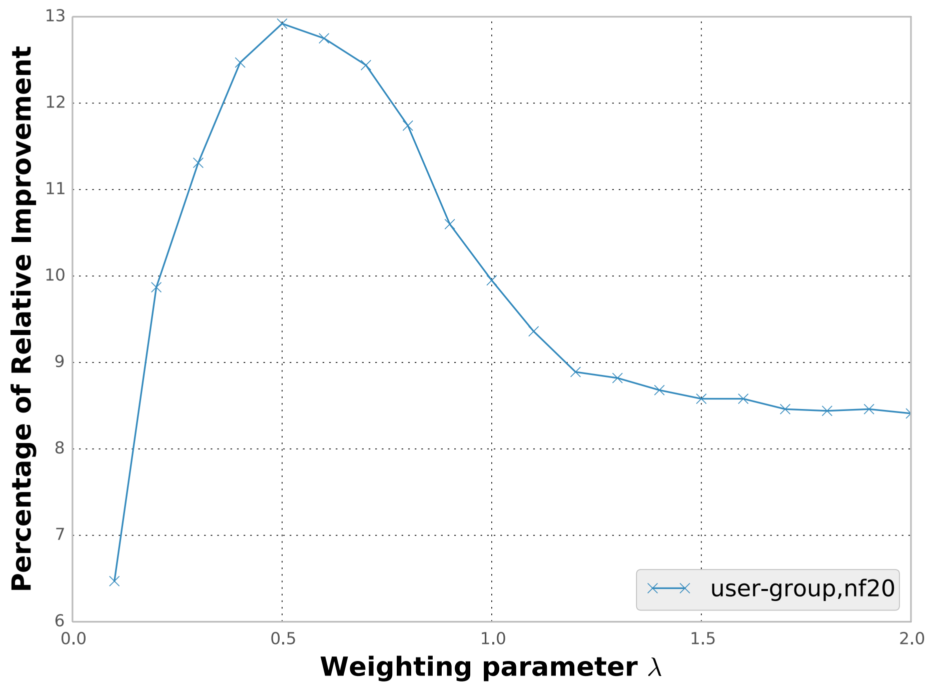
After learning the individual impact of *self-relationship*, we then combine different types of them for a task. We evaluate the user-group tasks with the help of S^U and S^G. In this setting, λ_U and λ_G are used to adjust the relative impact. Table 3.8 shows the comparison of S^U and S^G. From the results, combining both *self-relationship* does not further improve the tasks. To gain more insight, Fig 3.10 shows the contour plot of improvement with varifying both λ_U and λ_G. The plot shows the two local maximums locate both on the edges, when either one of them is zero. Therefore, add more both *self-relationship* does not gain improvement over individual *self-relationship*.

Table 3.8. Comparison of *WRS* with S^U and S^G.

self-relationship	user-group
S^U	6.42%
S^G	3.78%
S^U+S^U	6.44%

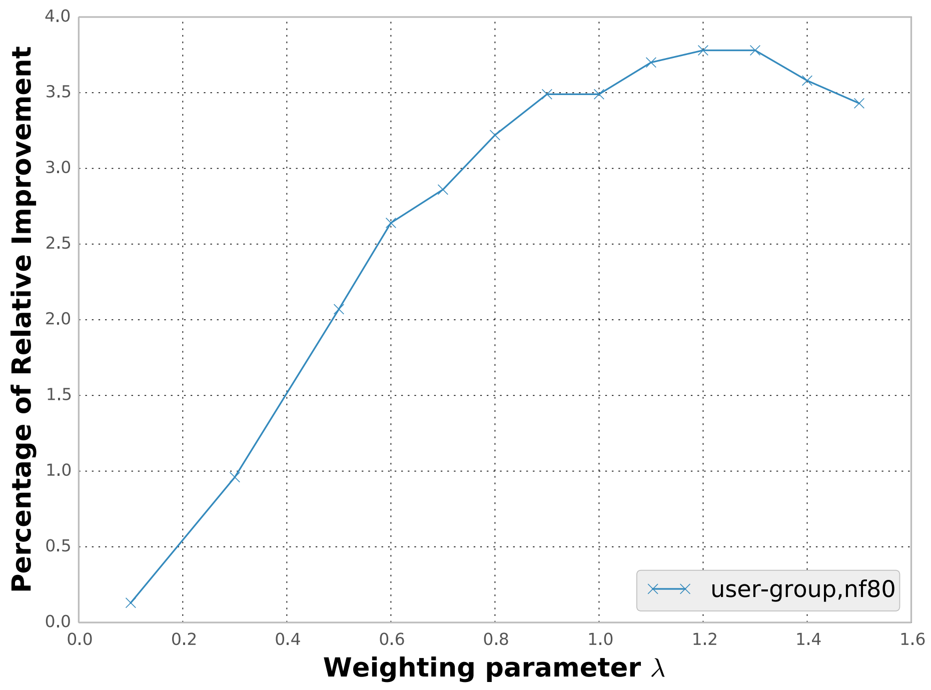


(a) *CV*

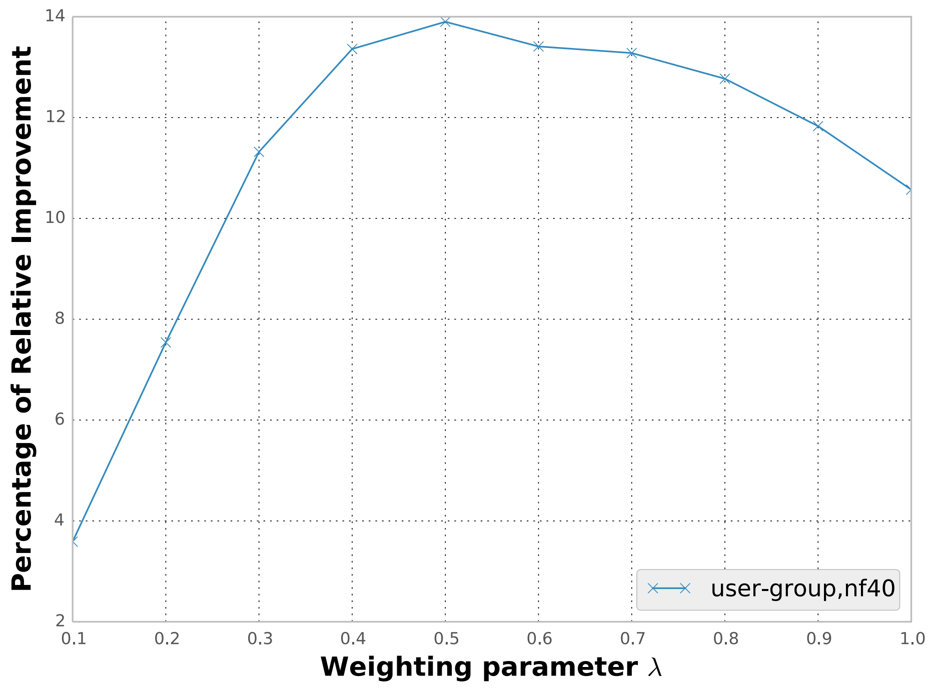


(b) *Given 20*

Figure 3.9a. Impact of λ with S^U on user-group.



(a) *CV*



(b) *Given 20*

Figure 3.9b. Impact of λ with S^G on user-group.

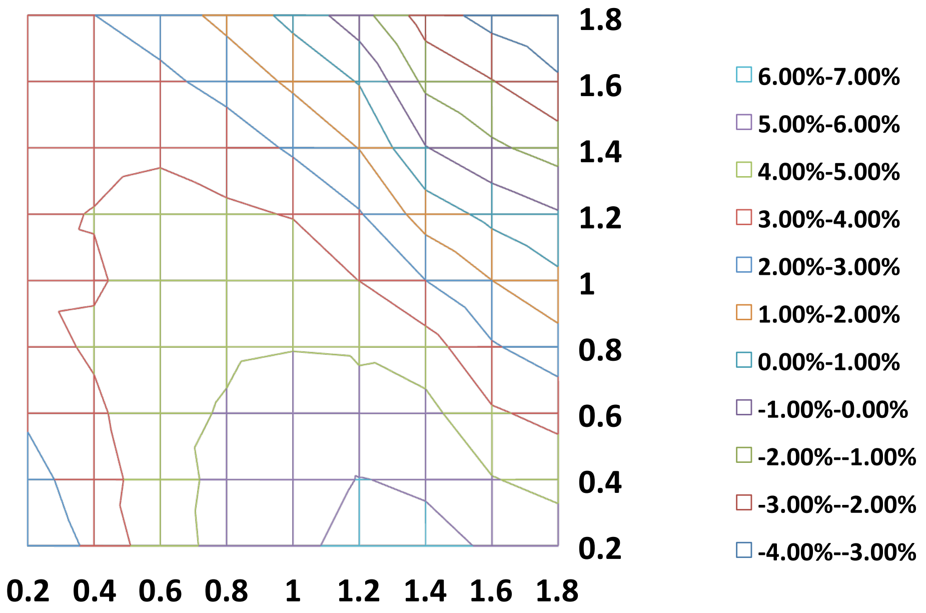


Figure 3.10. Selecting λ_U and λ_G.

3.4.4 Results of *WRI*

Next we present the results of *WRI* and make comparison of both *WRI* and *WRS*. Table 3.9 shows selected results of *WRI* with *CV* split, in which *WRI* further improves the results from *WRMF*, especially for user-image and image-group. For *Given 20*, as shown in Table 3.10, the improvements are significant for all three tasks. The overall performance is impacted by λs that balance the importance of different tasks. When a task has higher relative weight, the better the performance, and the lower the performance of the others. Table 3.11 shows the selection of different λ for user-group in *CV*. As the value increases, the performance of user-group gets better, while the performances of the other two tasks decline. Table 3.11 shows the selection of different λ for user-group in *Given 20*, which also demonstrates similar pattern.

Table 3.9. Results of *WRI* in *CV*.

K \Tasks	user-group	user-image	image-group
10	0.81%	8.36%	17.33%
20	1.27%	9.30%	12.61%
40	1.87%	5.68%	10.99%
80	-0.35%	10.23%	5.27%

Table 3.10. Results of *WRI* in *Given 20*.

K \Tasks	user-group	user-image	group-image
10	0.72%	1.48%	2.63%
20	2.75%	3.31%	19.68%
40	8.12%	13.67%	8.20%

Table 3.11. Varifying λ of *WRI* in *CV*.

λ \Tasks	user-group	user-image	group-image
1.2	0.11%	9.79%	13.12%
1.3	0.75%	8.71%	13.23%
1.4	1.27%	9.30%	12.61%
1.5	1.49%	9.52%	11.28%
1.6	2.53%	8.16%	9.83%
1.7	2.86%	7.00%	8.82%
1.8	3.16%	6.33%	7.79%
1.9	3.57%	5.60%	7.91%
2.0	4.05%	3.90%	5.71%

Table 3.12. Varifying λ of *WRI* in *Given 20*.

λ \Tasks	user-group	user-image	group-image
2.0	2.68%	0.01%	22.90%
2.2	2.75%	3.31%	19.68%
2.4	2.23%	4.80%	14.43%
2.5	2.50%	5.28%	11.34%
2.6	3.55%	5.99%	8.55%

Finally, we compare results from different proposed methods to *WRMF*. Table 3.13 shows the comparison for *CV*. For user-group, *WRS* with S^U has the best performance in terms of MRR, NDCG, and Recall@20 while *WRS* with S^U and S^G tops at Prec@20. For user-image and group-image, *WRI* clearly outperforms others. For *Given 20*, as shown in Table 3.14, *WRS* with S^U has best performance in user-group, while *WRI* outperforms other methods for the other two tasks.

Table 3.13. Comparison of Proposed Methods, *CV*, $K = 40$.

(a) user-group

Methods \Metrics	MRR	NDCG	Recall@20	Prec@20
WRMF	0.3904	0.4309	0.1220	0.1347
WRS (S^U)	**0.3967**	**0.4347**	**0.1251**	0.1360
WRS (S^G)	0.3930	0.4315	0.1229	0.1352
WRS (S^U,S^G)	0.3942	0.4346	0.1249	**0.1366**
WRI	0.3851	0.4305	0.1221	0.1345

(b) user-image

Methods \Metrics	MRR	NDCG	Recall@20	Prec@20
WRMF	0.0675	0.2213	0.0187	0.0154
WRS (S^U)	0.0682	0.2232	0.0196	0.0155
WRS (S^I)	0.0670	0.2212	0.0187	0.0154
WRI	**0.0738**	**0.2278**	**0.0212**	**0.0175**

(c) group-image

Methods \Metrics	MRR	NDCG	Recall@20	Prec@20
WRMF	0.1360	0.2632	0.0422	0.0459
WRS (S^G)	0.1367	0.2630	0.0414	0.0457
WRS (S^I)	0.1365	0.2634	0.0417	0.0459
WRI	**0.1408**	**0.2738**	**0.0459**	**0.0476**

3.5 Conclusion

In this Chapter, to full exploit the inter-connecitons between different subjects involved in a social network, we present a view of a social networks, where an entity has *self-relationship* with others of the same type and has *inter-relationship* with others of different types. According to same characteristics principle, we propose a co-factorization framework where each type of entity shares a common latent space. We proposed *WRS* to exploit the *self-relationship* , and *WRI* to exploit *inter-relationship* . We studied three types of *self-relationship* , including friendship network, textual similarity network for groups and images. In the experiments, we show the additional input from relationships can improve the recommendation further. However, combining multiple relationships might not gain extra improvements. Overall, the user-group recommenda-

Table 3.14. Comparison of Proposed Methods, *Given 20*, $K = 20$.
(a) user-group

Methods \Metrics	MRR	NDCG	Recall@20	Prec@20
WRMF	0.5811	0.5699	0.0604	0.2793
WRS (S^U)	**0.6303**	**0.6013**	**0.0696**	**0.3230**
WRS (S^G)	0.6104	0.5709	0.0639	0.3032
WRI	0.6164	0.5876	0.0648	0.3083

(b) user-image

Methods \Metrics	MRR	NDCG	Recall@20	Prec@20
WRMF	0.1324	0.3589	0.0091	0.0332
WRS (S^U)	0.1424	0.3695	0.0104	**0.0406**
WRS (S^I)	0.1305	0.3577	0.0094	0.0349
WRI	**0.1430**	**0.3751**	**0.0108**	0.0393

(c) group-image

Methods \Metrics	MRR	NDCG	Recall@20	Prec@20
WRMF	0.3019	0.4614	0.0306	0.1360
WRS (S^G)	0.3017	0.4622	0.0313	0.1363
WRS (S^I)	0.2962	0.4636	0.0305	0.1350
WRI	**0.3248**	**0.4804**	**0.0339**	**0.1457**

tion tasks is benefited with the friendship network, while user-image and group-image get improved with *inter-relationship* .

Chapter 4

Improving Classification Using User Generated Videos

4.1 Introduction

With the rapid development of technologies in software and hardware, users can now easily produce videos and share them in many social networks such Youtube, Facebook and so on. One popular form of these recorded videos is monologue, where users expressed themselves in speech in order to disseminate their ideas. Recently, many educational videos are created in this forms and are popularly shared, such as lecture videos and conference presentations. This particular types of videos, often referred as spoken documents, contains richer information in the speeches than other types and presents an unique opportunity for analysis with the techniques developed for textual contents. In this chapter, we explore the capability of textual mining with imperfect transcript from spoken documents. Compared to link-based content, the transcripts are more informative and compatible with text mining techniques. However, they often contain errors and their effectiveness as features for the mining task is still in doubt. Therefore, as first step to exploit the multimedia data toward uncover user status, we aim to study the capabilities of classification with noisy transcript by itself in this chapter. In order to compare the performance, we study performance from both noisy and perfect transcripts. In additional to the classification, our also study the keyword recommendation problem in the noisy transcript. Existing methods usually focus on 1-gram keyword extraction. However, sometimes, n-gram keyphrases (e.g., bi-gram like "black hole") could indicate hidden topics of documents better than 1-gram keywords (e.g., either "black" or "hole") does. Therefore, how to extend existing keyword extraction to n-gram keyphrases extraction is a challenge.

4.1.1 Problem Definition

4.1.1.1 Video Classification as Text Categorization Problem

Since the core intuition of our proposal is to treat the extracted transcripts of academic videos as documents and apply conventional text categorization techniques to these documents, in this section, we cover three popular text categorization methods that will be compared in our experiments (see Section 4.4.1).

The task of *text categorization* [37, 38] assigns a Boolean value to each pair $\langle d_j, c_i \rangle$ belonging to $D \times C$, where D is the domain of documents and $C = \{c_1, c_2, \ldots, c_N\}$ is the set of predefined categories. A value of T (true) assigned to $\langle d_j, c_i \rangle$ indicates that a document, d_j, is under a category, c_i, while a value of F (false) indicates otherwise. In other words, the text categorization task is to approximate the unknown target function $R' : D \times C \rightarrow \{T, F\}$, by means of a function $R : D \times C \rightarrow \{T, F\}$ called the *classifier* such that R' and R coincide as close as possible.

4.1.1.2 Keyword Extraction

The **key terms** of a video refer to representative n-gram words that capture the main topics or ideas of a video. Furthermore, we separately refer to a 1-gram and n-gram ($n \geq 2$) key terms as a **keyword** and **keyphrase**, respectively. The key terms can further facilitate users to efficiently search and browse target contents and help indexing, summarization, classification, and clustering [39].

Despite the extensive investigation, we believe that existing studies still lack of two aspects: (1) n**-gram:** Existing methods usually focus on 1-gram keyword extraction. However, sometimes, n-gram keyphrases (e.g., bi-gram like "black hole") could indicate hidden topics of documents better than 1-gram keywords (e.g., either "black" or "hole") does. Therefore, how to extend existing keyword extraction to n-gram keyphrases extraction is a challenge. (2) **Spoken Documents:** Existing methods tend to focus on the keyword extraction on *written* documents. However, the *spoken* documents which are

Table 4.1. An example of noisy spoken documents.

Noisy transcript
the idea was introduced in the context of <u>that after he dropped</u> carbon <u>we</u> don't need <u>to trade a</u> carbon in order to have handedness an example <u>shows is shown</u> by the oxidation <u>abbas</u> sulfide to <u>solve oxide</u>
Correct transcript
the idea was introduced in the context of a tetrahedral carbon but you don't need a tetrahedral carbon in order to have handedness and an example is shown by the oxidation of a sulfide to a sulfoxide

transcribed by automatic speech recognition (ASR) in VDL have a substantial degree of "errors" (as illustrated in Table 4.1) and present another challenge. The effectiveness of keyphrases extraction on spoken documents has not been investigated fully. In this paper, taking consideration with these two issues, we present our preliminary results of study on the effectiveness n-gram keyphrase extraction from noisy spoken documents.

While 1-gram keywords are, in general, good representatives for a given spoken document, when they are polysemous, they lack specificity, causing much confusion. In such a case, using an extra word could improve the specificity. For example, using "water bank" or "commercial bank" helps to clarify what the "bank" refers to. Furthermore, one thirds of human annotations tend to be n-gram keyphrases. To enable this idea, therefore, we aim to extend the unsupervised tf-idf method (that [40, 41] reported to be the best) to support keyphrase extraction as well.

The tf-idf weighing is a simple statistic scheme to identify words of high frequency in pivot documents that do not appear frequently in the whole corpus. Let d be a document in a corpora D, t is a term in d, whose frequency is tf_t. Then the tf-idf score of t is defined as: tf-idf$_t = tf_t \times \log(\frac{|D|}{|D_t|})$, where $|D|$ is the number of documents in D and $|D_t|$ is the number of documents in D that have the term t.

4.2 Related Work

4.2.1 Automatic Video Classification

Automatic video classification has been an active research area in recent year. The video classification tries to classify a given video into one of the predefined categories. The categories usually have conceptual or semantic meanings, such genres, subjects or topics. For example, [42] classifies movies by its genre: comedy, horror, action and drama/other movies, while [43] categorizes news videos into topics including politics, daily events, sports, weather, entertainment, health, business, and science& technology. A similar concept of video classification is the task of high-level feature extraction from TRECVID, in which each participant is required to return a list of shots in the corpus for each of the concepts, including indoor, outdoor, people, landscape, cityscape, etc.

The decision of a classification methods depend on the features of a video, including *audio*, *visual* and *textual* features. Researches may use only one of them or a combination of them, which is usually referred to as multi-modal or fusion-based approach.

1. *Audio* features can be extracted from the audio channel and can be used to help classification through different levels of audio understanding. For example, the audio signature of the sound of a bouncing ball or a song playing in the background may be identified and help to make classification. For the audio signature, zero-crossing rate (ZCR) [44] in the wave of audio and discrete cosine transform (DCT) [45] in frequency domain can be extracted and used for the classification method.

2. *Visual* features are very commonly used, which is often analyzed with cinematic principles. For example, for action movies the editor often changes shots frequently in a short period to make a perception of a fast rhythm to viewers. The examples of such visual features include colors [46] and motions [47]. However, most of academic videos have only speeches without dominant sound in the audio channel, so it would be difficult to connect the sound itself to any academic subjects. Moreover, academic videos usually have a small number of moving objects with little movement. The range of the camera motion and shot changes are also limited. These monotonic visual features would be also of little help for our classification task.

3. *Textual* features can be viewable text, transcript or metadata of the video. Such viewable text information are identifiable text data either filmed or added in post-production in the video, such as license number on car plate or scores in a sport game. The viewable text filmed in a video can be obtained by an optical character recognition (OCR) process [48]. In addition, transcript which is the dialog in a video can be obtained in their closed caption (CC) [43] or subtitle, or be extracted by automatic speech recognition (ASR) software. CC and subtitle may not always available while ASR can always be extracted if the video contains a speech (e.g., news or academic videos). However, as mentioned in Section 4.3.1, the accuracy of ASR against academic videos is much poorer than against regular videos. Metadata of the video are descriptive information associated with the video. The example includes file name, speaker name, title, length of video, etc. While some academic videos in the **Leedeo** project provide valuable metadata, large portion of them do not have any useful metadata, which makes video classification based on metadata difficult.

Qi et al. [49] use audio and visual features to segment news videos into stories, and then extract the text on the screen using OCR to classify the stories. Their method has around 80% accuracy in classifying 2 hours CNN news videos. Zhu et al. study the news story segmentation and classification with CC in [43]. They take advantage of the demarcations feature of CC to segment news videos and classify them with a weighted voting method. They evaluate the method by different percentages of training sample with 425 CNN news stories and 8 categories and achieve around 75% accuracy with 50% training samples. In [50], Lin et al. proposed a meta-classification strategy with SVM on CC and visual features from image to classify 440 CNN news stories into 2 categories, i.e., weather-report and others. Using CC alone it has 100% precision. Brezeale et al. classify 81 movies according to the movie genres, individual ratings user and grouped user ratings with CC as textual feature and DC terms as visual feature in [51]. From the result, CC consistently outperforms DC terms in all 3 tasks. In [52], Wang et al. propose a text-biased approach which mainly uses ASR transcript as textual. In addition, they use audio-visual feature as an extra clue to classify 559 CCTV news stories into 10 categories. Their methods use SVM on the ASR transcript, which only has around 10% error rate. From the result, the textual feature alone provides better precision than the combined features in most categories. Percannella et al. also classify news videos using SVM on ASR transcript in [53], and study the impact of using only parts of the news. Yang et al. report a study of classification of web-scaled videos by a two modalities approach which takes advantage of metadata of videos on the source web site [54]. The features used in this research include the titles, descriptions and tags of web videos on the web site and other 3 visual features. In their experiment, the textual features all outperform visual features and are comparative to the fusion of the features.

From the literatures, we conclude that the textual features have dominant impact on the semantic video classification, but most works on video classification with textual features focus on only news videos. Some studies report a better result when using multi-modal features but the improvement is usually limited at the cost of substantial time cost.

4.2.2 Spoken Document Retrieval and Classification

One notable earlier contribution on searching spoken document is the Spoken Document Retrieval (SDR) track from TERC 6 to TREC 9[55]. The goal is to investigate the retrieval methodologies on corrupted documents which are generated from automatic speech recognition and whose true content may not be available. On this evaluation-driven task, the participant implements a SDR system including two components: ASR and IR. For each topic in the test topics, the IR component returns a list of relevant documents in a collection of transcripts, whose true relevancy is assessed later by human judges. There are three tasks for the IR component: reference retrieval where a (nearly) perfect transcript is used, baseline retrieval where a ASR-generated prepared by NIST is used, and speech retrieval where ASR-generated transcripts participant's ASR component is used. In TREC 8, the dataset has 21,754 stories, 557 hours of news recordings from ABC, CNN, Public Radio International, and the Voice of America. The closed-captions of the recordings, though not perfect, are served as the perfect transcript. The error rate ranges from 7.5% (for radio) to 14.5% (for videos) in estimation. NIST provided two sets of baseline transcript B1 which has 27.5% WER and 2.54% OOV rate, and B2 which has 26.7% WER and 1.97% OOV rate. The best WER from participants' ASR component is 20.5%. A conclusion made from the result is that retrieval performance and word error rates have a near-linear relationship; however the performance degrades very gently (5%) for increasing recognition errors (10 35%). With the fact that the speech retrieval has a similar performance to reference retrieval with a large collection spoken documents, SDK is claimed as solved problem.

However, as pointed out in [56, 57], the recognizer in SDR is well-tuned for the new video, resulting in a low range of WER and OOV rate as low as 1.97%. Moreover, news speeches have many good properties that made the ASR task so successful. For example, it is usually prepared beforehand and professionally recorded, and its language style is close to written materials. There are many other spoken documents that don't have such properties, such as teleconferences or academic lectures, for which we may expect WER above 50%, a range that is not investigated in SDR. Recently, several methods are proposed to alleviate the problem by adding more possible words, such as N-best and lattice [58, 59], in the transcript to reduce WER, and by sub-word modeling, such as phonemes or syllables, to avoid OOV problem [60]. In particular, [61, 62] report experiment of retreival on the academic corpous. Alternately, [57] uses acoustic model adaption with discriminative acoustic modeling methods and language model adaptation learned from the related materials such as companion textbooks to reduce the WER from 33.6% to 28.4% on 6.1 hours of audio of 5 lectures given at MIT. Spoken Document Classification(SDC)is another fundamental task for understanding spoken documents. From the literature, it gained less attention than gained the SDR. Essentially, a SDC system connects the output of ASR to a Text Categorization (TC) module. The TC module extracts the features from the ASR-generated transcripts and performs classification with learning method. It is suspected that the classification performance would be affected by WER and OOV rate. As a result, several proposals are made with similar approaches to SDC, toward reducing the WER and OOV rate. [63] summarized the previous works into categories: the bag-of-words model, n-gram language model and sub-word model. In [64], instead of only use the frequency of term in bag-of-word model as the feature vector, the authors argument the vector with other linguistic sub-word units such as syllables, phonemes, and character n-grams. Using additional sub-word units ameliorates the fixed vocabulary problem while maintaining the semantic by words, at the cost of very large dimension vector. The paper also discuss the coupling problem in SVM for multiple-class classification and experiment their methods on TV and radio reports.

[65] compared the effect of classification with n-gram and different of sub-word (phonemes, syllables, words)on perfect transcripts and ASR-generated transcript. The test is conducted on 952 German radio programs, each of 5 minutes long, around 600

words and human-annotated in one of two subjects: politics and science. On the 1-gram ASR-generated syllables with only 30% accuracy, the result shows 6.7% and 48% drop of F1 measure for politics and science, respectively, from the 1-gram perfect syllables transcript. Overall, syllables and phonemes performs better than words on for perfect transcript; however, the relationship is clear on ASR-generated transcripts.

[63] proposed a phonotactic-semantic approach which is a mix of n-gram model and sub-word model to reduce OOV problem while achieving multilinguality and semantic abstraction. In their bag-of-sounds model, a spoken document is transcribed into a phonetic transcript which is a essentially document of sound tokens, and TC is applied on the transcript accordingly. This approach capture the sound characteristics of possibly multiple spoken languages by a universal collection of acoustic segment models without imposing any phonetic definitions. The method is evaluated with SVM and LSA on the 1996 NIST Language Recognition Evaluation (LRE) datasets which is original used for spoken language identification problem. The test set includes 1,492 30-sec sessions on 12 languages that are also served as class labels. The result shows 18.2% reduction in error over the benchmark performance.

4.2.3 Keyword Extraction

Supervised key term extraction has been well studied in the literature such as KEA [66], which is one of the state-of-the-art keyphrase extraction methods, developed in the New Zealand Digital Library project. Despite promising performance, the supervised framework needs time-consuming labeling process by domain experts. On the other hand, unsupervised extraction has been shown some promising results recently. Liu et al. [67] studied the keyword extraction problem on meeting transcripts. Their work showed that unsupervised extraction shares a similar performance with supervised extraction on meeting transcripts. Hasan et al. [40] compared five state-of-the-art unsupervised key term extraction methods (i.e. tf-idf, TextRank, SingleRank, ExpandLink and Clustering-based approach), and found that the simplest tf-idf based approach outperforms all the others on various corpora, including news articles, journal abstracts, conference papers and meeting transcripts.

In addition, many useful features are only provided on the well structure of a document. HaCohen-Kerner et al. [68] consider the position of a word in a sentence, the position of a word in a paragraph, and resemblance of a sentence to the title to select keywords for scientific articles. Kim et al. [69] take advantage of the syntactic structure of a sentence to extract the keyphrases in a scientific articles. However, these promising deep linguistic features are not available for the spoken documents transcribed by ASR.

In addition to keywords, keyphrases sometimes could indicate hidden topics in documents better. Liu et al.[41] conducted a user study where users freely annotate any word or phrase as key term. The final set of user annotation, it is reported that, consists of 66% of keywords and 33% of keyphrases. Chen et al.[70] reported a similar measure of 38% keyphrases of human annotation in their study. These results indicate that the use of keyphrases is substantial in human annotations. With the above observation, therefore, we aim to extend the study of unsupervised tf-idf based extraction [40] to keyphrase extraction and validate its effectiveness on erroneous spoken documents.

4.3 Proposed Methods

4.3.1 Classification Methods

The first step in text categorization is to deform the original document to the feature space. A set of features of a document are represented by a high dimensional feature vector, and each element of a vector is from a selected token in a document. In order to obtain proper tokens from documents, we apply stem-process and reduce the unnecessary large dimension of a feature. Then, we apply stop-process to remove stop-words such as *a*, *the*, or *and*.

In this study, we exploit three well known classification methods, i.e., Naive Bayes, KNN, and SVM, based on supervised learning approach. In other words, with labeled documents (i.e., the belongings of the categories are known), the proper classifier is acquired using learning methods or algorithms by obtaining the classification function of R. Note that these methods are known as good text classifiers, but the performance has not been verified as *spoken* document classifiers, which contains many noisy words from spoken-language or error-prone speech transcribers.

Spoken documents in our setting (i.e., extracted transcripts from academic videos) are almost always very *noisy* due to several reasons: (1) Since academic videos are more domain specific than regular videos (e.g., news or sports videos) are, often, vocabularies in academic videos are more peculiar and technical. Therefore, the accuracy of ASR software drops significantly for academic videos; and (2) Since the majority of academic videos are still produced in non-professional environment, the quality of audio in them is much poorer than are the videos professionally produced one, i.e., broadcasting companies. Such poor quality of transcript causes the accuracy of ASR software significantly to degrade when it is applied to academic videos. Therefore, it is not entirely clear whether conventional text categorization techniques would work well for spoken documents as well. We aim at answering this question in this section.

A naive Bayes classifier is a simple probabilistic classifier based on Bayesian theorem and is especially appropriate when the dimension of feature space is high [38]. Despite its simplicity, Naive Bayes can often outperform sophisticated classification methods. The probability of a document d being in class c is computed as:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c) \tag{4.1}$$

where $P(t_k|c)$ is the conditional probability of token t_k occurring in a document of class c and n_d is the total number of tokens. To find the most probable class in text categorization, we compute *maximum a posteriori* (MAP) class c_{map}:

$$c_{map} = \arg\max_{c \in C} \hat{P}(c|d) = \arg\max_{c \in C} \hat{P}(c) \prod_{1 \leq k \leq n_d} \hat{P}(t_k|c) \tag{4.2}$$

Since the parameter $P(c)$ and $P(t_k|c)$ in the equation (4.1) is estimated by training sets, we use $\hat{P}$ instead of P. $\hat{P}(c)$ is simply computed by $\frac{N_c}{N}$, where N_c is the number of documents in class c and N is total number of documents. $\hat{P}(t_k|c)$ can be estimated as the relative frequency of token t in documents belonging to class c such as $\frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$, where T_{ct} is the number of occurrences of t in training documents from class c. Since some tokens may not appear in certain classes, we may have *zero* in the equation (4.2). Thus, *Laplace smoothing* is applied to $\hat{P}(t_k|c)$ such that they become: $\frac{T_{ct}+1}{\sum_{t' \in V} T_{ct'}+1}$. For the case of academic spoken language, specially obtained by automatic speech recognition (ASR) transcriber, many inadequate words such as *hm* or *ah* are commonly found and thus removed during a preprocessing step. Therefore, the naive Bayes classifier can be directly applied to the noisy transcripts, because the key words will be kept with highly weighed value.

K-nearest neighbor (KNN) is one of the simplest machine learning algorithms [71]. A document is classified by the majority of K closest neighbors. For example, if $K = 1$, the unknown document will be classified by the known nearest neighbor (or class). KNN consists of two steps like other supervised algorithms. In the training step, the KNN classifier selects training set of documents and sprays them to the appropriate positions of the multi-dimensional field for the proper uses in the classification step. The unknown test documents are classified by K nearest known neighbors positioned in the training step. We use Euclidean distance in our feature space, i.e., $distance_i = ||d_i - d_0||$, where d_0 is a test document and d_i is a training document.

The support vector machine (SVM) classifier finds linear hyperplane boundaries in input feature space [72]. In this section, we start from two-class model and then expand it to the multi-class model.

If training data consists of N pairs (x_1, y_1), (x_2, y_2), ..., (x_N, y_N), with $x_i \in R^p$ and $y_i \in \{-1, 1\}$, the hyperplane is defined by $\{x : f(x) = x^T\beta + \beta_0 = 0\}$, where β is a unit vector, i.e., $||\beta|| = 1$. In addition, the classification rule induced by $f(x)$ is $G(x) = \text{sign}[x^T\beta + \beta_0]$. To obtain the biggest margin, in the case of overlapped classes (i.e., not separable), by defining the slack variables

$$\min ||\beta|| \text{ subject to } \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \zeta_i, \forall_i \\ \zeta_i \geq 0, \sum \zeta_i \leq \text{constant} \end{cases} . \tag{4.3}$$

Since the problem in Equation (4.3) is quadratic with linear inequality constraints, it becomes a convex optimization problem. With Lagrange multipliers, Equation (4.3) is re-expressed by $\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 + \gamma \sum_{i=1}^{N} \zeta_i$, where γ replaces the constant. Note that the separable case corresponds to $\gamma = \infty$. To acquire lower boundary, we need to maximize the Lagrangian dual object function, L_D=$\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i\, y_{i'}\, x_i^T\, x_{i'}$, subject to $0 \leq \alpha_i \leq \gamma$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$, where $\alpha = \{\alpha_i, \ldots, \alpha_N\}$ is Lagrange multipliers.

Generally, the support vector machine extends the dimension of the original feature space to achieve linear boundary separations in the enlarged space, corresponding to non-linear boundaries in the original feature space, using selected kernel function,

$K(x, x') = \langle h(x), h(x') \rangle$. As a result, the decision function can be re-written as:

$$\hat{G}(x) = \text{sign}\left[\sum_{i=1}^{N} \hat{\alpha}_i y_i K(x, x_i) + \hat{\beta}_0\right]. \tag{4.4}$$

Note that the details of induction procedure were omitted, because of space limit.

As we discuss earlier, SVM is subject to find hyperplane boundaries for 2-class classification. However, in many cases, we should classify more than 2 classes. For example, in our experiments, 17 classes exist in data sets. Thus, we compute the number of SVM one-by-one bases, hence the total number of SVM is $\frac{17(17-1)}{2} = 136$. For test data, we apply all decision functions and the majority of selected classes is decided as the belonging of input data.

4.3.2 Heuristic Keyphrase Extraction

The challenge to extract keyphrases using tf-idf based method lies on the fact that the frequency of keyphrases is usually much smaller than that of keywords. Therefore, often, deciding the importance of a keyphrase purely based on its tf-idf scores is inaccurate. In particular, note that tf-idf approach does not take the importance of sub-terms into consideration. Two keyphrases, say “commercial bank” and “water bank”, would have the same importance if they have the same tf-idf score (i.e., both appeared k times in the corpus). However, individually, if the sub-term “commercial” has a higher tf-idf score than “water” has, then “commercial bank” should bear a higher importance than “water bank” would have. In addition, to make matters complicated, we observed that if context is clear in a document, people tend to use shorter words instead of repeating the whole keyphrase. For example, in an article regarding “normal distribution”, the author may simply use “the distribution” to refer the full phrase.

Based on above observations, we proposes to extend the tf-idf based keyphrase extraction to incorporate the sub-terms’ scores, thus taking the importance of sub-terms into consideration. However, simply boosting the score of the phrases by its sub-terms’ score may lead to a set of very similar keyphrases extracted. For instance, a term “bank” with a high tf-idf score may appear in many keyphrases such as “commercial bank”, “water bank”, “bank loan”, etc. Therefore, over-boosting all these keyphrases’ score due to the “bank” may push all keyphrases in high ranks, leading the skewed keyphrase extraction. Therefore, we have to adjust weighting for different n-gram keyphrases and iteratively discount the weights of sub-terms from the already-extracted keywords.

Formally, let $T = \{t_1, t_2, ..., t_m\}$ be a candidate set of key terms, where each t_i (= $w_{i,1}w_{i,2}...w_{i,n}$) is an n-gram key term candidate, and $s(t_i)$ is the tf-idf score of the term t_i. Let t_1="probability distribution", then $w_{1,1}$="probability", and $w_{1,2}$="distribution". Our extraction runs in two steps:

1. **Boost**. We compute the normal tf-idf scores for all t's sub-terms. Then we boost their score as follows:

$$s_{new}(t) = \sum_{k=1}^{n} \alpha_k \sum_{j=1}^{n-k+1} s(w_j w_{j+1} ... w_{j+k-1})$$

   where α_k are the weight for k-gram terms. After the boosting, we select the top ranked n-gram keyphrase and remove it from the candidate set T.

2. **Discount**. For the keyphrases selected in step 1, we discount the scores of all its sub-terms, and redo the calculation to select the next keyphrase. β_k is a discounting parameter to decide how much the original score of a k-gram term should be diluted.

We repeat both boost and discount steps until the termination condition is met (e.g., the desired number of key terms are found). The details of the steps are shown in Algorithm 4.

4.4 Experimental Validation

4.4.1 Video Classification

4.4.1.1 Experimental Setup

The Naive Bayes (NB) and KNN algorithms are implemented in Matlab 7.0, while SVMlight [73] is used for the SVM implementation. All experiments were conducted on a HP Desktop with Intel Quad-core 2.4GHz, 4G RAM, and Windows Vista OS.

Data Set. To experiment our proposal, ideally, we need a data set that have sufficient number of video clips with various subjects. Moreover, to understand the impact of

Algorithm 4: Boost & Discount

```
input  : $l$, number of keywords to be selected.
output: $R$, set of keywords
repeat
    for $t$ in the candidate set $T$ do
        $s(t) \leftarrow 0$
        for $k = 1$ to $n$ do
            $kgram\_scr \leftarrow 0$
            for $j = 1$ to $n - k + 1$ do
                $kgram\_scr \leftarrow kgram\_scr + s(w_j w_{j+1} ... w_{j+k-1})$
            end
            $s(t) \leftarrow s(t) + \alpha_k \times kgram\_scr$
        end
    end
    add $t'$ that has the highest score to $R$
    remove $t'$ from $T$
    for $k = 1$ to $n$ do
        for $i = 1$ to $n - k + 1$ do
            $s(w_1 w_2 ... w_{i+k-1}) \leftarrow \beta_k \times s(w_1 w_2 ... w_{i+k-1})$
        end
    end
until $l$ keywords are selected;
```

noise in transcripts, the data set should have human-transcribed *perfect-quality*[1] transcripts. As a raw data set that meets these requirements, we chose the `Open Yale Courses` project[2] that provides a collection of introductory course videos and transcripts taught by faculty at Yale University. The collection includes 25 courses from 17 subjects. The number of lectures in each subject ranges from 20 to 37 while the length of lectures in each subject ranges from 35 to 90 minutes. The total running time is over 585 hours. Table 4.2 summarizes the statistics of the Yale data set. The Yale data set provides different formats of media for browsing such as video, audio, and human-transcribed transcripts.

Speech Recognition. The work of our speech recognition is based on Sphinx 4.0, one of the state-of-the-art hidden Markov model (HMM) speech recognition systems which

[1]While it is possible for human-transcribed transcripts may still have errors, compared to automatically generated ones, we believe their quality must be much more superior. Therefore, in this experiment, we consider human-transcribed transcripts as "perfect" ones.

[2]`http://oyc.yale.edu/`

Subject	# Courses	# Lectures	Length(hh:mm)
astronomy	1	24	18:48
bio-engineering	1	25	19:49
chemistry	1	37	30:04
classics	1	24	28:29
ee-biology	1	36	27:28
economics	2	26,24	59:34
english	4	26,26,24,25	87:05
history	3	27,24,24	60:25
history-of-art	1	23	27:36
italian-lang-lit	1	24	28:20
mcd-biology	1	24	27:02
music	1	23	19:01
philosophy	1	26	20:42
physics	1	24	28:43
political-sci	1	24	17:35
psychology	2	20,23	44:56
religious-studies	2	26,24	39:43
Total	**25**	**633**	**585:20**

Table 4.2. Statistics of the open Yale courses data set.

is a open source project from CMU [74]. In this pluggable framework of speech recognition, three main components exist, i.e., FrontEnd, Linguist, and Decoder. In our experiment, we use WordPruningBreadthFirstSearchManager as the search manager in the decoder and LexTreeLinguist as the Linguist. We adopt HUB4 acoustic model that have been trained using 140 hours of 1996 and 1997 hub4 training data which are continuous speeches from broadcast news [75]. We also use HUB4 language model which is a trigram model built for tasks similar to broadcast news. The vocabulary includes 64000 words.

Synthesized Test Sets. From the original Yale data set, next, we synthesize multiple test data sets for our experiments as follows:

(1) *Cross Validation*: To divide the data set into training and testing sets, two splits are considered: 5-fold *cross validation (CV)* and *course split (CS)*. First, in the CV test set, for each subject, randomly chosen 80% of transcripts are used for training, while remaining 20% for testing. This CV process repeats 5 times, and we measure the average at the end. Second, in the CS test set, for those subjects with more than 1 course (e.g., `English` and `History`), one course is chosen as the training set while another as the

testing set. While CV test sets aim to test whether subject can be determined given part of the lectures of a course, CS test sets aim to test whether subject can be determined given a complete new course that has not been studied for a classifier. Therefore, in general, CS test sets are more difficult than CV test sets.

(2) *Term Weighting*: Text categorization methods take a term matrix as an input, which is extracted from transcripts with various term weighting schemes. There are 4 term weighting schemes used in our experiments, i.e., `txx`, `tfx`, `txc` and `tfc` (using the notations from SMART system [76]), where `t`, `f`, `c`, and `x` indicates raw term frequency, inverse document frequency, cosine normalization, and none, respectively. For each transcript, we prepare 4 versions with `txx`, `tfx`, `txc` and `tfc` weighting schemes. The size of the overall term matrix is 31,762 terms and 634 documents after Porter's stemming.

(3) *Quality of Transcripts*: To compare the impact of the quality of transcripts in video classification, we compare three test sets: (i) a perfect transcript by human scribers (PF), (ii) an erroneous transcript with synthetic errors by Edit operations (i.e., insert, delete, and substitute) (SE), and (iii) an erroneous transcript by ASR software (SR). The quality of transcript is estimated by *Word Error Rate (WER)*, which is based on minimum edit distance of the erroneous transcript relative to a perfect one [77] defined as: $\text{WER} = \frac{\text{\# of insert+delete+substitute}}{\text{\# of words in the perfect transcript}}$.

(4) *Length of Transcripts*: Finally, full version *vs.* abbreviated test sets of transcripts are examined to see the impact of the length of transcripts for classification. For the abbreviated versions, we use the simple scheme of the first 10% of the first 10 minutes of total length of each transcript.

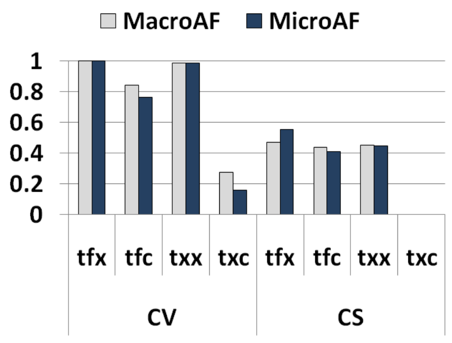
Evaluation Metrics. For the evaluation metrics, we consider F-measure in two popular averaging metrics to calculate the result of our multi-label classification task, i.e., macro-averaged F-measure (MacroAF), and micro-averaged F-measure (MicroAF). The macro/micro-averaged F-measure is determined by the macro/micro-averaged precision and recall. To calculate macro-averaged precision (MacroAP), the precision for each class is firstly calculated separately and then the MacroAP is calculated by taking the average of precisions across all classes. On the other hand, in micro-averaged precision (MicroAP), each transcript is given an equal weight so the MicroAP is the average precision of the data set as a whole. Formally, the averaging metrics are defined as:

$$\text{MacroAP} = \frac{1}{|K|}\sum_{i=1}^{|K|}\frac{TP_i}{TP_i + FP_i}, \text{MicroAP} = \frac{\sum_{i=1}^{|K|} TP_i}{\sum_{i=1}^{|K|} TP_i + FP_i}$$

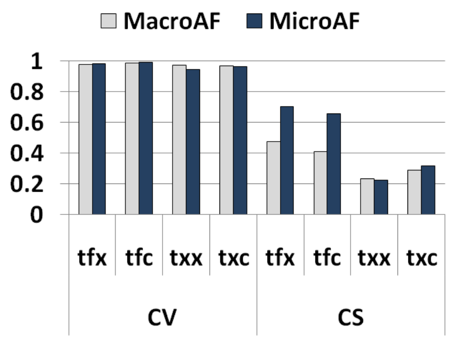
where TP_i and FP_i are the number of true positive and false positive for class i, respectively. The macro/micro-averaged recall (MacroAR/MicroAR) are defined in a similar fashion. Finally, MacroAF and MicroAF are defined as the harmonic mean of MacroAP and MacroAR, and that of MicroAP and MicroAR, respectively.

4.4.1.2 Result of Perfect Transcript

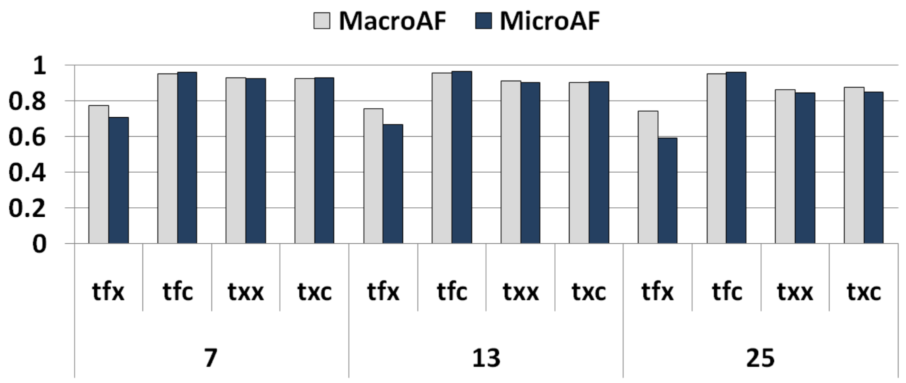
First, we assess the result of the baseline test, i.e., subject classification using *perfect* transcripts. Figure 4.1 (a) shows the baseline result of Naive Bayes (NB) method. For CV test set, the `tfx` weight scheme has the highest MacroAF and MicroAF (i.e., around 0.99), while the `txc` has the lowest MacroAF and MicroAF (i.e., 0.06 and 0.16, respectively). For much harder CS test set, overall F-measure drops by 40% from CV case. Among weighting schemes, it appears that `t` (term frequency) and `f` (inverse document frequency) play the major role, while somehow `c` (cosine normalization) works against. Note that in the CS test set, if two courses A and B are significantly different in their contents even if both belong to the same subject (say two courses "Computer Vision" and "Relational Databases" in Computer Science), then training using one course does not necessarily help in testing the other course. Therefore, in general, we expect a sharp drop in precisions when we compare CV to CS case. Second, the baseline result for the SVM method (with linear kernel and $C = 1$) is shown in Figure 4.1(b). All 4 term schemes have good F-measure ranging from 0.92 to 0.99. The F-measure for `tfx` and `tfc` against CS test set are around 0.6, while those of `txx` and `txc` are below 0.3. Overall, for CV test set, SVM method shows a promising result, especially compared to the other two text categorization methods. Finally, Figures 4.1(c)–(d) show the baseline result of the KNN method with K = 7, 13, and 25. All term schemes have F-measures exceeding 0.9, except for `tfx` weight scheme whose precision is around $0.6\% \sim 0.7\%$. For a much harder CS test set, F-measures rapidly drops compared to CV case. The best weighting schemes for CS case are `tfc` and `txx` which achieve F-measures around 0.4. On this baseline test, we observed that NB and KNN are more subject to the choice of



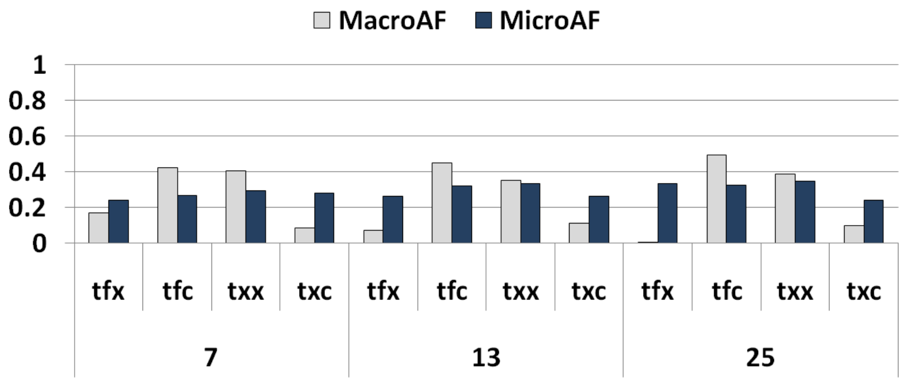
(a) Naive Bayes (NB)



(b) SVM



(c) KNN (CV)



(d) KNN (CS)

Figure 4.1. Precisions using perfect transcripts.

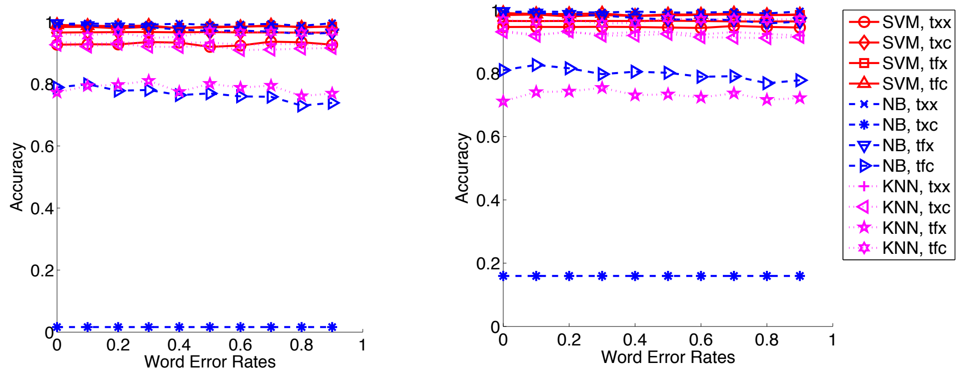
weight scheme, and that micro-averaging and macro-averaging have similar results on our dataset.

4.4.1.3 Effect of the Noise of Transcript

Although there are a large number of academic videos available on the Web, the availability of their perfect transcripts are very limited since human transcription is an expensive and time-consuming task. Meanwhile, automatic speech recognition (ASR) software can generate imperfect machine-transcribed transcripts at decent speed and cost. At the time of the writing, however, the word error rate (WER) of the state-of-the-art ASR system is around 10% for broadcast news and around 20% for conversational telephone speech as reported in [77]. Moreover, as discussed in Section 4.3.1, when ASR system is applied to academic videos, its WER increases even further due to poor audio quality and peculiar vocabularies used. While conventional text categorization methods perform relatively well (especially for CV test sets) in Section 4.4.1.2, we wonder if that remains true with "noisy" transcripts which are more commonly encountered in practical settings. However, an ASR recognizer interacts with several models and the complexity limits the possibility of predicting its behavior. Instead of building a model of possible output from an ASR recognizer, we take a approach of simulation to study relation of different levels of noisy transcripts and classification performance in term of precision.

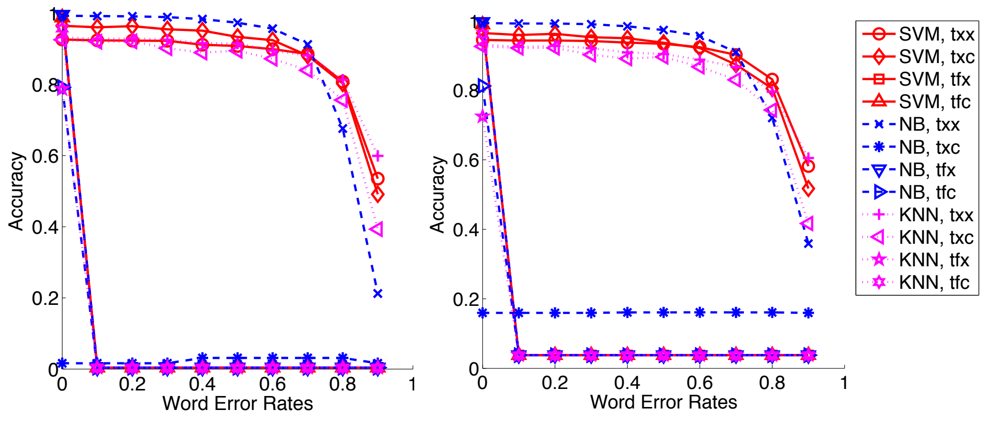
Synthetically Erroneous Transcripts (SE). The synthetically erroneous transcript allows us to simulate different levels of WERs and the impact of edit operations therein. In our experiment, 3 different edit operations (i.e., `add`, `delete`, and `substitute`) are simulated independently on all the perfect transcripts as follows: (1) *Addition*: a number of terms for specific error rate are selected in a uniformly random fashion and their term frequency is added by one. A term can only be selected once; (2) *Deletion*: a fixed number (100 by default) of terms whose frequency are greater than zero are selected in a uniformly random fashion, and their term frequency is deducted by one accordingly. This procedure repeats until the total number of desirable deletions is achieved; and (3) *Substitution*: we performs aforementioned deletion followed by the insertion.

The MacroAF and MicroAF for insertion are shown in Figure 4.2. The SVM method performs well and remains stable for all 4 term weighting schemes. The NB has the best precision with `txx` and `tfx` but the performance with `tfc` slightly decreases when WER goes high. Again, `txc` does not work well in this task. Precision for KNN (K=7) ranges from 0.7 $\sim$ 0.9 except with `tfc` which is around 0.8. In conclusion, when



(a) MacroAF (b) MicroAF

Figure 4.2. Precision of SE test set with addition.

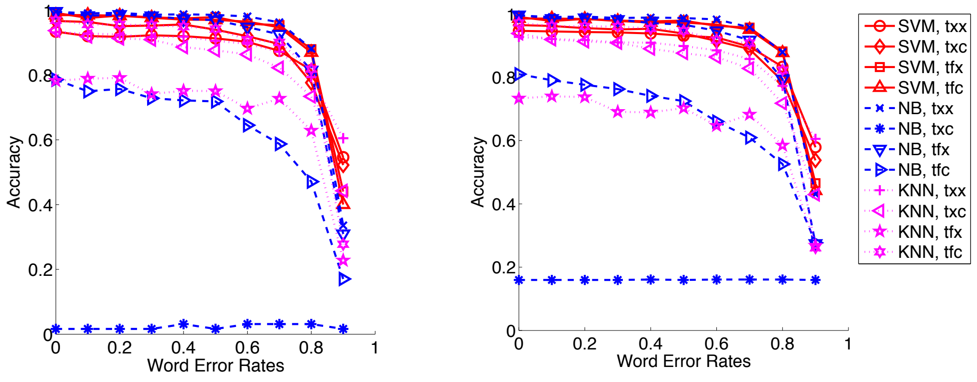


(a) MacroAF (b) MicroAF

Figure 4.3. Precision of SE test set with deletion.

transcripts have errors due to insertion operations, the performance changes depending on the chosen weighting schemes, but it remains robust even if WER increases. Both MacroAF and MicroAF have very similar pattern.

Figure 4.3 shows the result with deletion. For SVM, only `txx` and `txc` work well initially but their precisions decrease as WER increases, especially WER exceeds 0.6. The results of KNN are similar to those of SVM but a little worse for all 4 weighting schemes. NB has only `txx` working, whose precision drops sharply after WER of 0.7. Figure 4.4 shows the result for substitution. When WER is low, the results are similar to the results of insertion for 4 weighting schemes. The performance slightly decreases as WER increases and drops sharply after WER of 0.8.



(a) MacroAF (b) MicroAF

Figure 4.4. Precision of SE test set with random substitution.

4.4.1.4 Effect of the Length of Transcripts

In this section, we investigate the impact of length of the transcript to the video classification. We test 3 cases on the length of transcript including the first 1%, 5%, and 10% of perfect transcript and ASR-generated transcript. The intuition to shorten the length of transcripts by taking the first 1–10% is that in many academic videos such as lecture or presentation videos, speakers often give a summary of the talk in the beginning. On average, the length of the first 10% of perfect transcripts and the first 10 minutes of ASR-generated transcripts is around 600 and 1000 words, respectively.

Figure 4.5 shows a comparison of all three methods on different length of perfect transcripts with CS grouped by weighting schemes. In Figure 4.5(a), `tfx` scheme shows the best accuracy, i.e., using only the first 1% of transcripts, NB could achieve the F-measure of 0.9. In Figure 4.5(b), `tfc` scheme which has the best performance, shows roughly the same results using the 100%, 10%, and 5% of transcripts, but has a sharp drop from 5% to 1%. The F-measure of other schemes also decrease, but have larger differences when moving from 100% to 10% and to 5%. In Figure 4.5(c), SVM has high F-measure regardless of the chosen weighting scheme. This is similar to the results of `tfc` with KNN and `tfx` with NB. We believe if a method (with a specific weighting scheme) has a strong precision and recall, the length of transcripts used in the classification has less impact on them. Even with, 1% of the length, it is still possible to achieve the F-measure of 0.8. On the other hand, above a threshold of certain length, the F-measure becomes stable.

Figure 4.6 shows the result of using the different length of ASR-generated transcript. For NB, `tfx` shows the best result in which MicroAF reaches 0.81 with the first 5% transcripts. MicroAF reaches 0.92 with the first 10% transcript, which is comparable to the result on the first 10% perfect one, 0.95. For KNN, only `tfc` can be considered effective, whose F-measure is 0.5 with the first 1% of transcript, and improves to 0.80 with the first 10%. For SVM, all weighting schemes show similar performance. The F-measures are between 0.5 0.6 on the first 1% transcript, and improve to 0.8 0.9 on the first 5% one.

From this experiment, we observed that, with a good choice of term weighting, the result of the first 5% of full length is comparable to that of full length transcript. For ASR-generated transcript, we need at least the 10% to reach similar performance. Moreover, we may conclude that for: (1) The text categorization is susceptible to the choice of term weighting scheme; (2) The CS test set is more difficult to classify and may need be studied further; and (3) The impact of the quality and length of transcript exists, but not substantial. Validated with the CV test set, all three text categorization methods can achieve the F-measure over 0.8 with a good choice of weight scheme. However, in general, SVM shows the most robust performance.

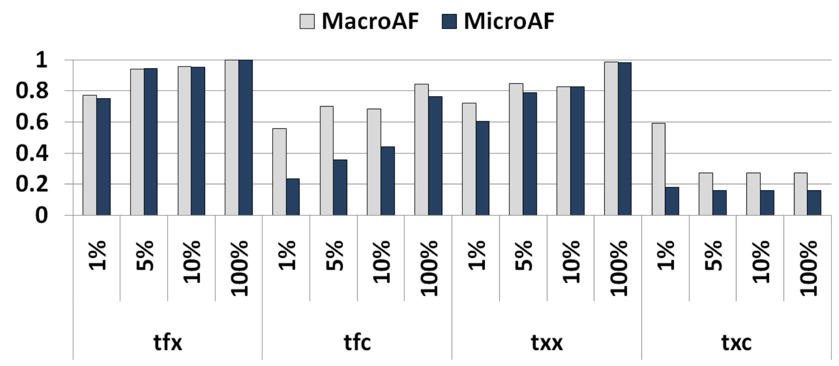
4.4.2 Keyword Extraction

4.4.2.1 Experimental Setup

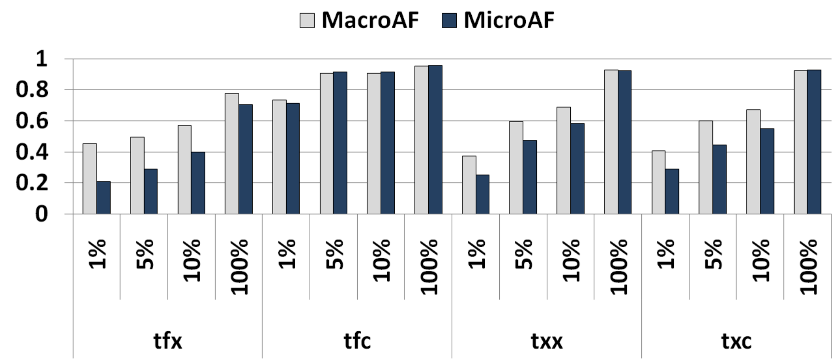
Data Set. We used Open Yale Courses collection[3] as the corpus. The collection consists of hundreds of lecture recordings taught in Yale University. It includes dozens of subjects from 20 different departments, making both ASR and key term extraction tasks challenging. For each lecture, the collection provides a near-perfect human-transcribed transcript (i.e., *Reference*), used as a ground truth. Then, for each lecture, we prepared ASR-generated transcript (i.e., *Hypothesis*) using Sphinx 3[4] with acoustic and language models trained with the HUB4 news data set. Table 4.3 shows basic statistics of the subjects of selected lectures.

Before the extraction task, we preprocessed both references and hypotheses with stop words removal, part-of-speech filtering, and lemmatization. While keyphrase can

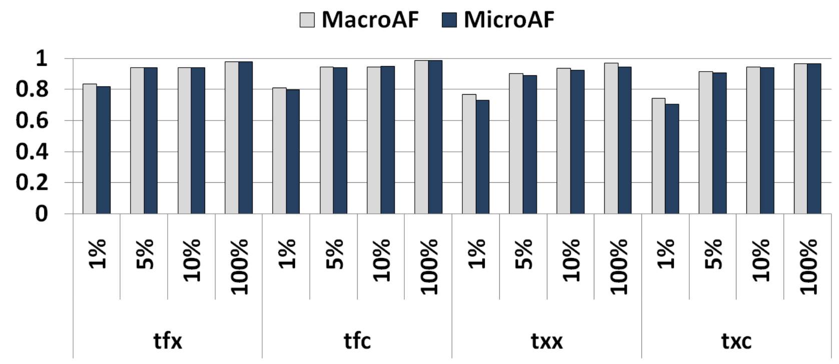
[3]`http://oyc.yale.edu/`
[4]`http://cmusphinx.sourceforge.net/`



(a) Naive Bayes (NB)



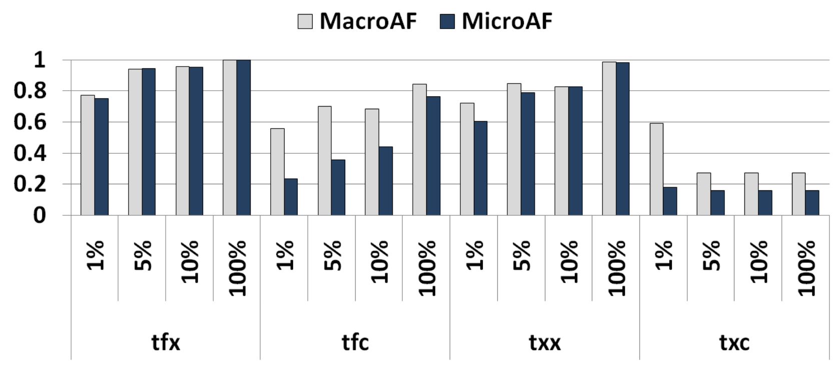
(b) KNN (K=7)



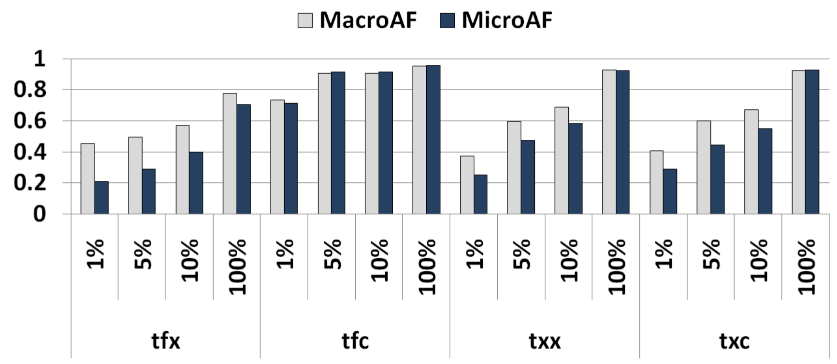
(c) SVM

Figure 4.5. Comparison using different length of perfect transcripts on CV.

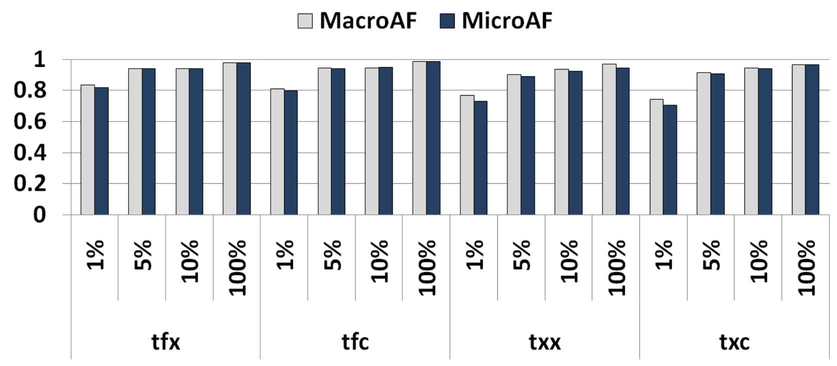
include any n-gram terms with $n \geq 2$, in this experiment, we focused on the comparison of 1-gram (i.e., keyword) and 2-gram terms (i.e., keyphrase) since the frequency of 3-grams (and beyond) is comparatively low in our corpus after preprocessing. The user study aims to compare the performance of original tf-idf extraction and our proposed method on both types of transcripts. We randomly selected one lecture from each sub-



(a) Naive Bayes (NB)



(b) KNN (K=7)



(c) SVM

Figure 4.6. Comparison using different length of ASR-generated transcripts on CV.

ject in the collection and had three users review it. The average *word error rate* (WER[5]) estimation for hypothesis is 45.78%. Note that the WER is relatively high since many courses contain challenging conditions for ASR–e.g., cross languages, multiple speakers, poor recording quality, etc. However, the average *correct words percentage* (CWP[6])

[5]WER = $100 \times \frac{\text{Insertions+Substitutions+Deletions}}{\text{Total Words in Reference}}$

[6]CWP=$100 \times \frac{\text{Words Matched in Edit Distance Alignment}}{\text{Total Words in Reference}}$

Table 4.3. Statistics of the subjects of selected lectures.

Subjects	Ref. Len.	Hyp. Len.	WER (%)	CWP (%)
astr	7220	7468	31.48	75.72
beng	6989	7470	28.56	80.70
chem	8716	8920	54.51	55.79
clcv	9975	10611	37.39	72.01
econ	9827	9040	60.81	49.45
eeb	6717	9995	37.20	69.11
engl	6209	6830	42.87	70.46
hist	6771	7614	52.77	64.94
hsar	12200	13095	42.62	68.21
ital	9569	10864	75.70	41.11
mcdb	9906	9982	45.19	61.85
musi	5656	6383	44.55	70.77
phil	7727	7742	38.02	67.09
phys	11624	11157	65.84	38.82
plsc	5791	6215	33.33	75.70
psyc	10537	10595	37.95	67.76
rlst	8670	9004	41.49	66.54
Mean	**8212**	**8646**	**45.78**	**64.84**

is 64.84%, more appropriate to represent the quality of hypotheses based on the bag of words model of tf-idf.

Evaluation. For 1-gram key terms, top 15 key terms are extracted with original tf-idf extraction on references (**ref-1g**), and hypotheses (**hyp-1g**). For 2-gram key terms, top 10 keywords are extracted with tf-idf extraction on references (**ref-2g-tdf**) and on hypotheses (**hyp-2g-tdf**), with our proposed method (i.e., Boost & Discount) on references (**ref-2g-bd**), and on hypotheses (**hyp-2g-bd**). We ran a preliminary test and manually checked the results to select the parameters α and β. Empirically, a uniform weighting for each term works well. Therefore the following values are used: $\alpha_k = \frac{1}{n(n-k+1)}$ and $\beta_k = \frac{1}{2}$. All key terms extracted with different settings are then pooled into 1-gram candidate set and 2-gram candidate set according to the number of grams, and sorted so that order has no significance. For each lecture to be judged, users are presented with its reference and two pooled sets of key term candidates. Users then pick one to five key terms that are most representative to the reference from the two candidate sets separately. The limitation on the number of term to be selected aims to test if the systems can generate the *best* (rather than just *relevant*) key terms. Note that this is a more

Table 4.4. An example of extracted key terms.

ref-1g	***hyp-1g***
probability	probability
finance	insurance
nala	finance
insurance	distribution
***ref-2g-bd* (our proposal)**	***ref-2g-tdf***
probability theory	random variable
random variable	probability theory
independent probability	interest rate
insurance company	black swan

Table 4.5. Comparison of keyphrase extraction.

Top 10 Keyphrases				
settings	ref-2g-bd	ref-2g-tdf	hyp-2g-bd	hyp-2g-tdf
# of selection	4.21			
# of terms from	**3.05**	2.55	**2.47**	1.97

restrictive condition and will generally reduce the precision. As the human judges, 15 graduate students with science or engineering majors were recruited.

Many previous studies used a set of human annotated key terms as a gold standard and compared the outputs from proposed extraction system against the gold standard to evaluate the performance. However, this method may not evaluate the precision accurately since the quality of the key terms is not directly judged. To test the effectiveness of extracted key terms and provide a fair ground for the comparison of different methods, therefore, we propose to use the TREC style pooling method [78], where top-ranked retrieved documents from all systems are "pooled" together and the relevance of each documents in the pool is judged individually by human judges. The main difference of the proposed evaluation to the gold standard method is that the judges evaluate the effectiveness of extracted key terms *directly* (instead of proposing their own key terms).

4.4.2.2 Original tf-idf vs. Boost & Discount

In the user study, a human judge is presented with a set of key term candidates which are top-10 outputs from four different settings. Table 4.4 gives an example of extracted key terms. We are interested in finding out the methods that contribute more number of key terms selected by human judge. The average size of the pool is 18.48, indicating

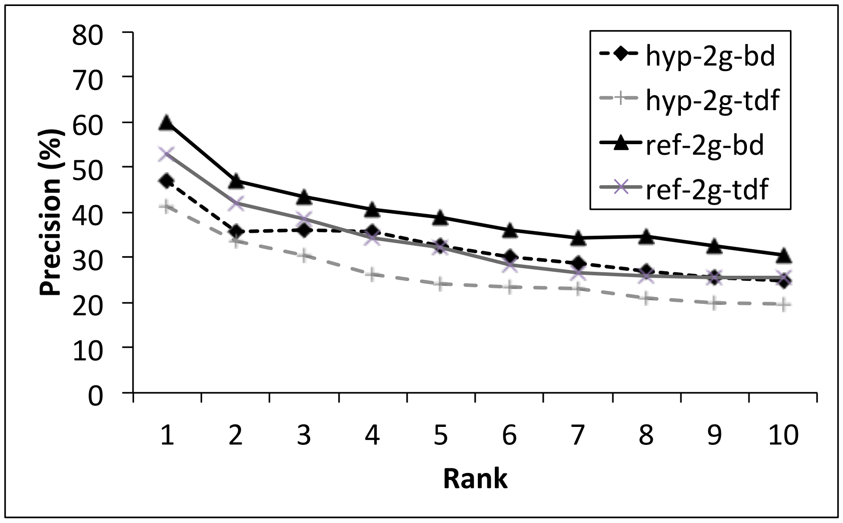


Figure 4.7. Impact of ranks on keyphrase extraction.

that those methods share a number of terms.

Table 4.5 shows the summary of what human judges selected. On average, judges selected 4.21 key terms. Among these selection, 1.97 are attributed to **hyp-2g-tdf** while 2.47 terms to **hyp-2g-bd**. That is, our proposed B&D method is about 25% more accurate over the tf-idf method using the hypotheses data set, with a statistically significant difference ($p < 0.01$). Our method also has an improvement of 19%, from 2.55 for **ref-2g-tdf** to 3.05 for **ref-2g-bd** ($P < 0.03$).

The impact of the ranking across different methods is also investigated, as shown in Fig 4.7, where X-axis is the number of rank used and Y-axis is the precision (i.e., the proportion of accumulated key terms selected by users). For example, if a method suggests 5 key terms (in top-5 rank) from **ref-2g-bd** and 2 of them are also selected by human judges, the precision becomes 40%. The figure shows that our B&D method outperforms the tf-idf extraction method over *all* the ranks using *both* types of transcripts.

The td-idf statistics is known to be susceptible if domains of documents vary [79]. Since our Yale data set contains a dozen different subjects, in Figure 4.8, we studied the impact of unsupervised extraction on different knowledge domains with **hyp-2g-bd** and **hyp-2g-tdf**. Each data point indicates the averaged precision from lectures under different subjects. As demonstrated, precision varies across different subjects. Overall, subjects such as engineering show better precision. Biomedical Engineering (beng), Ecology & Evolutionary Biology (eeb), and Molecular Cellular and Developmental Bi-

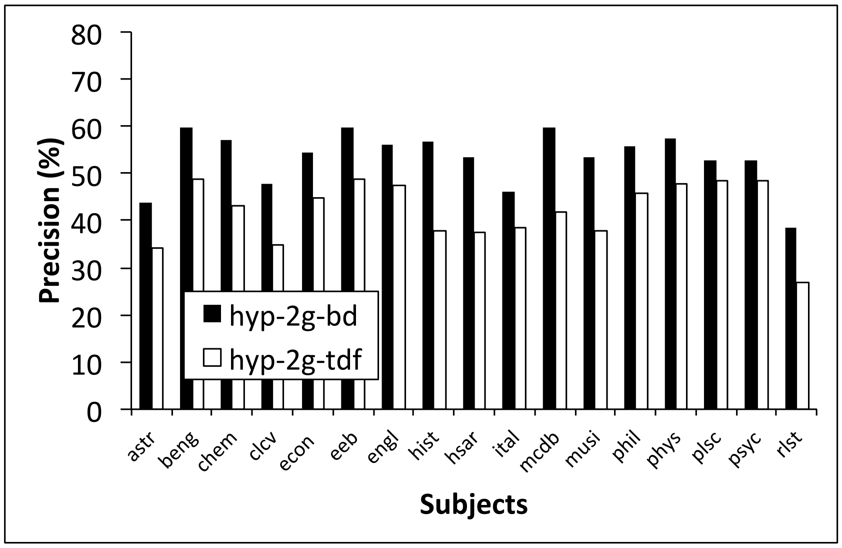


Figure 4.8. Precisions across different subjects.

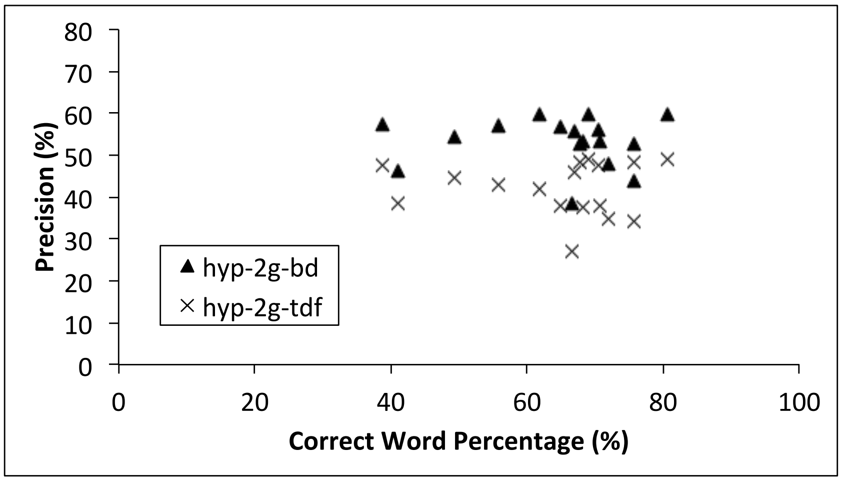


Figure 4.9. Precisions across different CWPs.

ology (mcdb) are subjects that showed the highest precision. Religious Studies (rlst) and Italian Language and Literature (ital), however, showed lower precisions, partly due a substantial use of non-English words in spoken documents. Finally, Figure 4.9 shows that our proposal shows improved precision over the tf-idf approach across all ranges of CWP.

Table 4.6. Comparison btw. hypothesis and reference.

Top 15 Keywords		
Settings	ref-1g	hyp-1g
# of selection	5.55	
# of terms from	4.28	4.16

4.4.2.3 Hypothesis vs. Reference

Next we study the impact of erroneous transcripts. First, we note that from the top 15 keywords extracted, **ref-1g** and **hyp-1g** have an average overlap rate of 52.53%, having about 7.88 number of terms in common. On average, 5.55 number of 1-gram keywords are selected per lecture. From the results, as shown in Table 4.6, we found 4.28 of them are from **ref-1g** and 4.16 are from **hyp-1g**. It shows that the 1-gram keyword extraction on hypotheses have very similar performance, though inferior but not significantly, to that on references. For the 2-gram keyphrases, as shown in Table 4.4, the overlap rate is 39.6% between **ref-2g-bd** and **hyp-2g-bd** on average among the top 10 key terms extracted. The result shows 2-gram keyphrase extraction on hypotheses has 19% degradation than that on references with the proposed method. The tf-idf method has 22% degradation. The larger degradation, however, is expected since 2-gram keyphrases are harder to be correctly recognized in ASR process. For instance, if "digital library conference " is recognized as "digital signal conference", then the CWP would be 66% for 1-gram terms but 0% for 2-gram terms.

4.5 Conclusion

We have conducted comparative study on the subjects classification of academic videos with various methods. For the study, we transformed the video subject classification problem into the text categorization one by exploiting the transcripts of videos. We also investigated the impact of plausible factors, such as noise/quality of transcripts and the length of a video with three popular text categorization methods. Our experimental results shows that SVM promises the best result in both CV and CS cases. In terms of *term weights*, `tfx` is good in NB and SVM, but not in KNN, while `tfc` is good in SVM and KNN, but not in NB. From the synthetically erroneous transcripts test, we observed that the learning methods with a good choice of weighting scheme is very robust even though 70% of the transcript is incorrect. In addition, the learning methods is dependable when only a small part of transcript, human or machine transcribed, are available.

We also proposed a heuristic improvement for the unsupervised keyphrase extraction by incorporating the importance of sub-terms in an iterated fashion, and validated its effectiveness with a pooling method. Our method shows 25% improvement from the original method on ASR-generated transcripts. The proposed method also made 19% improvement on reference transcripts. The user study shows that keyword extraction on erroneous ASR-generated transcripts is almost effective as that on references, with only 2.8% decrease of accuracy. Compared to the keyphrases from references, however, the effectiveness degrades to 19.3% for keyphrases due to the ASR difficulties.

Chapter 5

Conclusion

5.1 Conclusion

For the location estimation problem, we proposed a novel approach to estimate the spatial word usage probability with Gaussian Mixture Models. We also proposed unsupervised measurements to rank the local words to effectively reduce the noises that are harmful to the prediction. We show that our approach can, using less than 250 local words, achieve a comparable or better performance to the state-of-the-art that uses 3,183 local words selected by the supervised classification based on 11,004 hand-labeled ground truth. For the demographic status prediction problem, we showed that using text feature is effective in predicting various user statuses.

For the item recommendation task, the proposed new view of a social network accommodates rich relationships in a system. The proposed co-factorization framework based on same characteristics principle shares a common latent space for the same type of entities, thus effectively connecting different recommendation tasks, and side information. As a results, both *WRS* and *WRI* showed significant improvement over *WRMF*, indicating its effectiveness in exploiting *self-relationship* and *inter-relationship*, respectively. The data sparseness is also alleviated from the more substantial improvement with the *Given 20* split. However, we also learn that adding multiple relationships might not always gain extra improvements. Overall, the user-group recommendation tasks is benefited with the friendship network, while user-image and group-image get improved with *inter-relationship*.

In the study on classification of noisy textual content, we learn that the performances are comparable between perfect content and noisy content through extensive experiments with different classification methods. We also found that SVM yields most promising result for both CV and CS cases. As for the feature selection, `tfx` works well with NB and SVM, but not with KNN, while `tfc` works well with SVM and KNN, but not with NB. We also observed that the classification is in general very robust even with the presence of noisy content up to 70%. The learning methods is dependable when only a small part of transcript, human or machine transcribed are available. Finally, our proposed heuristic keyword extraction takes the importance of sub-terms into account. Evaluated with a pooling method, the results show 25% improvement over the original method with noisy content, and 19% improvement with perfect content. The user study shows the keyword extraction equally effective for noisy content as opposed to the perfect content. However, the effectiveness degrades to 19.3% for keyphrases due to the ASR difficulties.

Many directions are ahead for future work. For the location estimation problem, to further improve accuracy is one of them. Discriminative method tends to have better results than a generative method. Applying our local word selection with discriminative method might be another one. On the other hand, social relation has been shown to be effective in recent studies. Incorporation of additional features to the textual based ones might also help the performance. Furthermore, extending the current work to predict the user mobility is another interesting direction.

Bibliography

[1] LAMPOS, V., T. DE BIE, and N. CRISTIANINI (2010) "Flu detector: tracking epidemics on twitter," in *Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part III*, ECML PKDD'10, Springer-Verlag, Berlin, Heidelberg, pp. 599–602.

[2] PAUL, M. J. and M. DREDZE (2011) "You Are What You Tweet: Analyzing Twitter for Public Health," in *Proc. of the 5th International AAAI Conference on Weblogs and Social Media*, ICWSM'11.

[3] SANKARANARAYANAN, J., H. SAMET, B. E. TEITLER, M. D. LIEBERMAN, and J. SPERLING (2009) "TwitterStand: news in tweets," in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '09, ACM, New York, NY, USA, pp. 42–51.

[4] SAKAKI, T., M. OKAZAKI, and Y. MATSUO (2010) "Earthquake shakes Twitter users: real-time event detection by social sensors," in *Proceedings of the 19th international conference on World wide web*, WWW '10, ACM, New York, NY, USA, pp. 851–860.

[5] QAZVINIAN, V., E. ROSENGREN, D. R. RADEV, and Q. MEI (2011) "Rumor has it: identifying misinformation in microblogs," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1589–1599.

[6] CHENG, Z., J. CAVERLEE, and K. LEE (2010) "You are where you tweet: a content-based approach to geo-locating twitter users," in *ACM CIKM*, Toronto, ON, Canada, pp. 759–768.

[7] CHANDRA, S., L. KHAN, and F. B. MUHAYA (2011) "Estimating Twitter User Location Using Social Interactions-A Content Based Approach," in *IEEE SocialCom*, pp. 838–843.

[8] HECHT, B., L. HONG, B. SUH, and E. H. CHI (2011) “Tweets from Justin Bieber’s heart: the dynamics of the location field in user profiles,” in *ACM CHI*, Vancouver, BC, Canada, pp. 237–246.

[9] KINSELLA, S., V. MURDOCK, and N. O’HARE (2011) ““I’m eating a sandwich in Glasgow”: modeling locations with tweets,” in *Int’l workshop on Search and Mining User-Generated Contents (SMUC)*, Glasgow, Scotland, UK, pp. 61–68.

[10] BACKSTROM, L., E. SUN, and C. MARLOW (2010) “Find me if you can: improving geographical prediction with social and spatial proximity,” in *Proceedings of the 19th international conference on World wide web*, WWW ’10, ACM, New York, NY, USA, pp. 61–70.

[11] SADILEK, A., H. KAUTZ, and J. P. BIGHAM (2012) “Finding your friends and following them to where you are,” in *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM ’12, ACM, New York, NY, USA, pp. 723–732.

[12] DALVI, N., R. KUMAR, and B. PANG (2012) “Object matching in tweets with spatial models,” in *ACM WSDM*, Seattle, Washington, USA, pp. 43–52.

[13] LI, W., P. SERDYUKOV, A. P. DE VRIES, C. EICKHOFF, and M. LARSON (2011) “The where in the tweet,” in *ACM CIKM*, Glasgow, Scotland, UK, pp. 2473–2476.

[14] SERDYUKOV, P., V. MURDOCK, and R. VAN ZWOL (2009) “Placing flickr photos on a map,” in *ACM SIGIR*, Boston, MA, USA, pp. 484–491.

[15] VAN LAERE, O., S. SCHOCKAERT, and B. DHOEDT (2011) “Finding locations of flickr resources using language models and similarity search,” , pp. 48:1–48:8.

[16] KELM, P., S. SCHMIEDEKE, and T. SIKORA (2011) “Multi-modal, multi-resource methods for placing Flickr videos on the map,” in *ACM Int’l Conf. on Multimedia Retrieval (ICMR)*, Trento, Italy, pp. 52:1–52:8.

[17] LARSON, M., M. SOLEYMANI, P. SERDYUKOV, S. RUDINAC, C. WARTENA, V. MURDOCK, G. FRIEDLAND, R. ORDELMAN, and G. J. F. JONES (2011) “Automatic tagging and geotagging in video collections and communities,” in *ACM Int’l Conf. on Multimedia Retrieval (ICMR)*, Trento, Italy, pp. 51:1–51:8.

[18] ZHAI, C. and J. LAFFERTY (2004) “A study of smoothing methods for language models applied to information retrieval,” *ACM Trans. Inf. Syst.*, **22**(2), pp. 179–214.

[19] BACKSTROM, L., J. KLEINBERG, R. KUMAR, and J. NOVAK (2008) “Spatial variation in search engine queries,” in *WWW*, Beijing, China, pp. 357–366.

[20] SALTON, G. (1971) *The SMART Retrieval System - Experiments in Automatic Document Processing*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[21] MCPHERSON, M., L. S. LOVIN, and J. M. COOK (2001) "Birds of a Feather: Homophily in Social Networks," *Annual Review of Sociology*, **27**(1), pp. 415–444.

[22] PAN, R., Y. ZHOU, B. CAO, N. N. LIU, R. LUKOSE, M. SCHOLZ, and Q. YANG (2008) "One-Class Collaborative Filtering," in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, IEEE Computer Society, Washington, DC, USA, pp. 502–511.

[23] SU, X. and T. M. KHOSHGOFTAAR (2009) "A Survey of Collaborative Filtering Techniques," *Adv. in Artif. Intell.*, **2009**, pp. 4:2–4:2.

[24] KOREN, Y., R. BELL, and C. VOLINSKY (2009) "Matrix Factorization Techniques for Recommender Systems," *Computer*, **42**(8), pp. 30–37.

[25] HU, Y., Y. KOREN, and C. VOLINSKY (2008) "Collaborative Filtering for Implicit Feedback Datasets," in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, IEEE Computer Society, Washington, DC, USA, pp. 263–272.

[26] RENDLE, S., C. FREUDENTHALER, Z. GANTNER, and L. SCHMIDT-THIEME (2009) "BPR: Bayesian Personalized Ranking from Implicit Feedback," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, AUAI Press, Arlington, Virginia, United States, pp. 452–461.

[27] SHI, Y., A. KARATZOGLOU, L. BALTRUNAS, M. LARSON, N. OLIVER, and A. HANJALIC (2012) "CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering." in *RecSys* (P. Cunningham, N. J. Hurley, I. Guy, and S. S. Anand, eds.), ACM, pp. 139–146.

[28] SHI, Y., A. KARATZOGLOU, L. BALTRUNAS, M. LARSON, A. HANJALIC, and N. OLIVER (2012) "TFMAP: optimizing MAP for top-n context-aware recommendation." in *SIGIR* (W. R. Hersh, J. Callan, Y. Maarek, and M. Sanderson, eds.), ACM, pp. 155–164.

[29] LACERDA, A. and N. ZIVIANI (2013) "Building User Profiles to Improve User Experience in Recommender Systems," in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, ACM, New York, NY, USA, pp. 759–764.

[30] MA, H., I. KING, and M. R. LYU (2009) "Learning to Recommend with Social Trust Ensemble," in *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, ACM, New York, NY, USA, pp. 203–210.

[31] Jamali, M. and M. Ester (2010) "A Matrix Factorization Technique with Trust Propagation for Recommendation in Social Networks," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, ACM, New York, NY, USA, pp. 135–142.

[32] Ma, H., H. Yang, M. R. Lyu, and I. King (2008) "SoRec: Social Recommendation Using Probabilistic Matrix Factorization," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, ACM, New York, NY, USA, pp. 931–940.

[33] Ning, X. and G. Karypis (2012) "Sparse Linear Methods with Side Information for Top-n Recommendations," in *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, ACM, New York, NY, USA, pp. 155–162.

[34] Park, S., Y.-D. Kim, and S. Choi (2013) "Hierarchical Bayesian Matrix Factorization with Side Information," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, AAAI Press, pp. 1593–1599.

[35] Forsati, R., M. Mahdavi, M. Shamsfard, and M. Sarwat (2014) "Matrix Factorization with Explicit Trust and Distrust Side Information for Improved Social Recommendation," *ACM Trans. Inf. Syst.*, **32**(4), pp. 17:1–17:38.

[36] Li, Y., J. Hu, C. Zhai, and Y. Chen (2010) "Improving One-class Collaborative Filtering by Incorporating Rich User Information," in *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, ACM, New York, NY, USA, pp. 959–968.

[37] Sebastiani, F. and C. N. D. Ricerche (2002) "Machine Learning in Automated Text Categorization," *ACM Computing Surveys*, **34**, pp. 1–47.

[38] Manning, C. D., P. Raghavan, and H. Schutze (2008) *Introduction to Information Retrieval*, Cambridge Univ. Press.

[39] Jones, S. and G. W. Paynter (2002) "Automatic extraction of document keyphrases for use in digital libraries: evaluation and applications," *JASIST*, **53**, pp. 653–677.

[40] Hasan, K. S. and V. Ng (2010) "Conundrums in Unsupervised Keyphrase Extraction: Making Sense of the State-of-the-Art," in *COLING*, pp. 365–373.

[41] Liu, Y., S. Xie, and F. Liu (2010) "Using n-best recognition output for extractive summarization and keyword extraction in meeting speech," in *IEEE ICASSP*.

[42] Rasheed, Z. and M. Shah (2002) "Movie Genre Classification by Exploiting Audio-Visual Features of Previews," in *IEEE ICPR*, Quebec City, Canada, pp. 1086–1089.

[43] Zhu, W., C. Toklu, and S. Liou (2001) "Automatic news video segmentation and categorization based on closed-captioned text," in *IEEE ICME*, pp. 829–832.

[44] Wold, E., T. Blum, D. Keislar, and J. Wheaten (1996) "Content-based classification, search, and retrieval of audio," *Multimedia, IEEE*, **3**(3), pp. 27–36.

[45] Logan, B. (2000) "Mel Frequency Cepstral Coefficients for Music Modelingl," in *1st International Symposium on Music Information Retrieval (ISMIR)*.

[46] Gupta, A. and R. Jain (1997) "Visual information retrieval," *Communication of the ACM*, **40**(5), pp. 70–79.

[47] Iyengar, G. and A. Lippman (1998) "Models for Automatic Classification of Video Sequences," in *SPIE*.

[48] Hauptmann, A. G., R. Yan, Y. Qi, R. Jin, M. G. Christel, M. Derthick, M. Chen, R. V. Baron, W. Lin, and T. D. Ng (2002) "Video Classification and Retrieval with the Informedia Digital Video Library System," in *TREC*.

[49] Qi, W., L. Gu, H. Jiang, X. Chen, and H. Zhang (2000) "Integrating visual, audio and text analysis for news video," in *ICIP*, Vancouver, Canada, pp. 10–13.

[50] Lin, W. (2002) "News Video Classification Using SVM-based Multimodal Classifiers and Combination Strategies," in *MM*, ACM Press, pp. 1–6.

[51] Brezeale, D. (2006) "Using closed captions and visual features to classify movies by genre," in *Poster session of MDM/KDD*, Philadelphia, USA.

[52] Wang, P., R. Cai, and S. Yang (2003) "A Hybrid Approach to News Video Classification Multimodal Features," in *International Conf.on Info., Comm. and Signal Processing*.

[53] Percannella, G., D. Sorrentino, and M. Vento (2005) "Automatic Indexing of News Videos Through Text Classification Techniques," in *ICAPR (2)*, Lecture Notes in Computer Science, pp. 512–521.

[54] Yang, L., J. Liu, X. Yang, and X. Hua (2007) "Multi-modality web video categorization," in *MIR*, New York, NY, USA, pp. 265–274.

[55] Garofolo, J. S., C. G. P. Auzanne, and E. M. Voorhees (2000) "The TREC Spoken Document Retrieval Track: A Success Story," in *in Text Retrieval Conference (TREC) 8*, pp. 16–19.

[56] Chelba, T., C. Hazen and M. Saraclar (2008) "Retrieval and Browsing of Spoken Content," *IEEE Signal Processing Magazine*, **25**, pp. 39–49.

[57] GLASS, J., T. J. HAZEN, S. CYPERS, I. MALIOUTOV, D. HUYNH, and R. BARZILAY (2007) "Recent progress in the MIT Spoken Lecture Processing Project," in *Proceedings of Interspeech*.

[58] CHIA, T. K., K. C. SIM, H. LI, and H. T. NG (2010) "Statistical lattice-based spoken document retrieval," *ACM Trans. Inf. Syst.*, **28**(1), pp. 1–30.

[59] (2009) *Word-lattice based spoken-document indexing with standard text indexers*.

[60] NG, K. (2000) *Subword-based approaches for spoken document retrieval*, Ph.D. thesis, supervisor-Zue, Victor W.

[61] CHELBA, C., J. SILVA, and A. ACERO (2007) "Soft indexing of speech content for search in spoken documents," *Comput. Speech Lang.*, **21**(3), pp. 458–478.

[62] SEIDE, F., P. YU, and Y. SHI (2007) "Towards spoken-document retrieval for the enterprise: approximate word-lattice indexing with text indexers," in *in Proc. ASRU'07*, pp. 629–634.

[63] MA, B. and H. LI (2005) "A phonotactic-semantic paradigm for automatic spoken document classification," in *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, New York, NY, USA, pp. 369–376.

[64] IURGEL, U. and G. RIGOLL (2004) "Spoken Document Classification with SVMs Using Linguistic Unit Weighting and Probabilistic Couplers," vol. 2, IEEE Computer Society, Los Alamitos, CA, USA, pp. 667–670.

[65] PAASS, G., E. LEOPOLD, M. LARSON, J. KINDERMANN, and S. EICKELER (2002) "SVM Classification Using Sequences of Phonemes and Syllables," in *PKDD '02: Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, Springer-Verlag, London, UK, pp. 373–384.

[66] WITTEN, I. H., G. PAYNTER, E. FRANK, C. GUTWIN, and C. G. NEVILL-MANNING (1999) "KEA: Practical Automatic Keyphrase Extraction," in *ACM DL*.

[67] LIU, F., F. LIU, and Y. LIU (2008) "Automatic keyword extraction for the meeting corpus using supervised approach and bigram expansion," in *SLT Workshop*, pp. 181 –184.

[68] HACOHEN-KERNER, Y., Z. GROSS, and A. MASA (2005) "Automatic extraction and learning of keyphrases from scientific articles," in *Proceedings of the 6th international conference on Computational Linguistics and Intelligent Text Processing*, CICLing'05, Springer-Verlag, Berlin, Heidelberg, pp. 657–669.

[69] KIM, S. N. and M.-Y. KAN (2009) "Re-examining automatic keyphrase extraction approaches in scientific articles," in *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications*, MWE '09, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 9–16.

[70] CHEN, Y.-N., Y. HUANG, S.-Y. KONG, and L.-S. LEE (2010) "Automatic key term extraction from spoken course lectures using branching entropy and prosodic/semantic features," in *SLT Workshop*, pp. 265 –270.

[71] TAN, S. (2006) "An effective refinement strategy for KNN text classifier," *Expert Syst. and Appl.*, **30**(2), pp. 290–298.

[72] HASTIE, T., R. TIBSHIRANI, and J. FRIEDMAN (2001) *The Elements of Statistical Learning*, Springer.

[73] JOACHIMS, T. (1999) "Making large-Scale SVM Learning Practical," *Advances in Kernel Methods–Supp. Vector Learning*.

[74] WALKER, W., P. LAMERE, P. KWOK, B. RAJ, R. SINGH, E. GOUVEA, P. WOLF, and J. WOELFEL (2004) *Sphinx-4: a flexible open source framework for speech recognition*, *Tech. rep.*, Mountain View, CA, USA.

[75] PLACEWAY, P., S. CHEN, M. ESKENAZI, U. JAIN, V. PARIKH, B. RAJ, M. RAVISHANKAR, R. ROSENFELD, K. SEYMORE, M. SIEGLER, R. STERN, and E. THAYER (1996) "The 1996 Hub-4 Sphinx-3 System," in *In Proc. of DARPA Speech Recognition Workshop*, The.

[76] SALTON, G. (1971) *The SMART Retrieval System—Experiments in Automatic Document Proc.*, Prentice-Hall, NJ, USA.

[77] JURAFSKY, D. and J. H. MARTIN (2009) *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognitiong*, Prentice-Hall.

[78] HARMAN, D. (1995) "Overview of the Fourth Text REtrieval Conference," in *TREC*.

[79] TURNEY, P. D. (2002) *Mining the Web for Lexical Knowledge to Improve Keyphrase Extraction: Learning from Labeled and Unlabeled Data*, *Tech. rep.*, National Research Council, Inst. for Info. Tech.

VITA

Education

- PhD in Computer Science and Engineering, Penn State University, University Park, PA, Aug 2015
- M.S. in Applied Mathematics, National Chung-Hsing University, Taiwan, June 2003
- B.S. in Applied Mathematics, National Chung-Hsing University, Taiwan, June 2001

Work Experience

▪ 2012/09 – 2013/04	Research Co-op in Nittany System Research, State College, PA
▪ 2008 Summer	Intern at IBM Sunnyvale, CA
▪ 2010 – 2012	Research Assistant, in CSE, Penn State University, State College, PA
▪ 2007 – 2009	Teaching Assistant, in CSE, Penn State University, State College, PA

Publications

- @Phillies Tweeting from Philly? Predicting Twitter User Locations with Spatial Word Usage, Hau-Wen Chang, Dongwon Lee, Mohammed Eltaher, Jeongkyu Lee, In IEEE/ACM Int'l Conf. on Advances in Social Networks Analysis and Mining (ASONAM), Istanbul, Turkey, August 2012
- Comparative Study on Subject Classification of Academic Videos using Noisy Transcripts, Hau-Wen Chang, Hung-sik Kim, Shuyang Li, Jeongkyu Lee, Dongwon Lee, In IEEE Int'l Conf. on Semantic Computing (ICSC), Pittsburgh, PA, September 2010
- BASIL: Effective Near-Duplicate Image Detection using Gene Sequence Alignment, Hung-sik Kim, Hau-Wen Chang, Jeongkyu Lee, Dongwon Lee, In The 32nd European Conf. on Information Retrieval (ECIR), Milton Keynes, UK, March 2010
- BIM: Image Matching using Biological Gene Sequence Alignment, Hung-sik Kim, Hau-Wen Chang, Haibin Liu, Jeongkyu Lee, Dongwon Lee, In IEEE Int'l Conf. on Image Processing (ICIP), Cairo, Egypt, November 2009
- Adopting XML Technology for Network Management Information Processing, Master Thesis, National Chung Hsing University, 2003
- An Efficient Network Management System Based on XML Technology, Fu-Ming Chang, Hau-Wen Chang, and Shang-Juh Kao, In The 31st International Conference on Computers and Industrial Engineering. Limerick, Ireland August, 2003