Collaborative filtering (CF) is an effective technique addressing the information overload problem. CF approaches generally fall into two categories: rating-based and ranking-based. The former makes recommendations based on historical rating scores of items and the latter based on their rankings. Ranking-based CF has demonstrated advantages in recommendation accuracy, being able to capture the preference similarity between users even if their rating scores differ significantly. In this study, we propose VSRank, a novel framework that seeks accuracy improvement of ranking-based CF through adaptation of the vector space model. In VSRank, we consider each user as a document and her pairwise relative preferences as terms. We then use a novel degree-specialty weighting scheme resembling TF-IDF to weight the terms. Extensive experiments on benchmarks in comparison with the state-of-the-art approaches demonstrate the promise of our approach.

Categories and Subject Descriptors: H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—Information Filtering

General Terms: Algorithms, Performance, Experimentation

Additional Key Words and Phrases: Recommender systems, Collaborative filtering, Ranking-based collaborative filtering, Vector space model, Term weighting

ACM Reference Format:
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

Ever since the thriving of the Web, the world has been flooded with an overwhelming amount of information. Such a wealth of information has become increasingly unmanageable and created “a poverty of attention and a need to allocate that attention efficiently” [Simon 1971]. This so-called information overload problem represents one of today’s major challenges on the Web. As an effective technique addressing the problem, recommender systems generate item recommendations from a large collection in favor of user preferences. In recent years, recommender systems have become a de facto standard and must-own tool for e-commerce to promote business and help customers find
products [Sarwar et al. 2000]. Prominent examples include eBay\(^1\), Amazon\(^2\), Last.fm\(^3\), Netflix\(^4\), Facebook\(^5\), and LinkedIn\(^6\).

1.1. Collaborative Filtering

The two main paradigms for recommender systems are content-based filtering and collaborative filtering (CF). Content-based filtering makes recommendations by finding regularities in the textual content information of users and items, such as user profiles and product descriptions [Belkin and Croft 1992]. CF is based on the assumption that if users \(X\) and \(Y\) rate \(n\) items similarly or have similar behaviors, they will rate or act on other items similarly [Goldberg et al. 1992; Resnick et al. 1994].

CF only utilizes the user-item rating matrix to make predictions and recommendations, avoiding the need of collecting extensive information about items and users. In addition, CF can be easily adopted in different recommender systems without requiring any domain knowledge [Liu and Yang 2008]. Given the effectiveness and convenience, many CF methods have been proposed, which fall into two categories: rating-based and ranking-based.

**Rating-based CF.** Rating-based CF methods recommend items for users based on their historical rating scores on items. As a classical CF paradigm, they have been extensively investigated, where most methods are either memory-based [Resnick et al. 1994; Herlocker et al. 1999; Sarwar et al. 2001; Deshpande and Karypis 2004] or model-based [Vucetic and Obradovic 2005; Shani et al. 2005; Si and Jin 2003; Tang et al. 2013; Jiang et al. 2012].

Rating-based CF utilizes similarity measures between two users based on their rating scores on the set of common items. A popular similarity measure is the Pearson correlation coefficient [Resnick et al. 1994; Herlocker et al. 2002]. However, the ultimate goal of a recommender system is to present a ranking or recommendation list to users rather than rating prediction [Shi et al. 2010; Herlocker et al. 2004]. In addition, as a common observation, such rating-based similarity measures would fail to capture preference similarity between users when their rating scores on items differ significantly [Adomavicius and Tuzhilin 2005; Gunawardana and Shani 2009], as illustrated in the following.

**Example 1.1.** Let \(\{i_1, i_2, i_3\}\) be three items. Let \(\{u, v\}\) be two users who have assigned ratings of \(\{1, 2, 3\}\) and \(\{3, 4, 5\}\) to the items respectively. While \(u\) and \(v\) exhibit clear common relative preferences over the three items, their rating scores differ significantly, leading to small rating-based similarity between \(u\) and \(v\).

A straightforward normalization based on average rating scores of users does not address this problem effectively. This is because when a given user only rated a very small set of items, her average rating may differ significantly from her “true” (but unknown) average rating, which can be defined as the average value of her ratings on all potential items in the item space. For example, suppose \(u\) is a very generous reviewer, and her “true” average rating is 4. Suppose \(u\) has reviewed 5 low-quality items, and her average rating is 2. Thus an item with a rating of 3 will be interpreted as favorable by \(u\) after normalization based on the average rating of 2. However, the truth is that \(u\) dislikes the item because 3 is smaller than her “true” average rating of 4.

**Ranking-based CF.** Ranking-based CF methods recommend items for users based on their rankings of items. In particular, such methods utilize similarity measures between two users based on their

\(^1\)http://www.ebay.com/
\(^2\)http://www.amazon.com/
\(^3\)http://www.last.fm/
\(^4\)http://www.netflix.com/
\(^5\)http://www.facebook.fm/
\(^6\)http://www.linkedin.com/
rankings on the same set of items. A common similarity measure is the Kendall tau rank correlation coefficient [Kendall 1938; Marden 1995]. Recent efforts on ranking-based CF [Yang et al. 2011; Weimer et al. 2007; Liu and Yang 2008; Liu et al. 2009; Rendle et al. 2009; Cui et al. 2011; Shi et al. 2010; Kahng et al. 2011] have clearly demonstrated their advantages in recommendation accuracy.

However, conventional ranking-based CF algorithms treat pairwise relative preferences equally, without considering any weighting scheme for preferences in similarity measures. For example, relative preferences may have different degrees depending on how strong the preferences are. In addition, two users are considered similar if they share some special traits instead of common ones.

1.2. VSRank for Ranking-based Collaborative Filtering

The vector space model [Baeza-Yates and Ribeiro-Neto 1999] is a standard and effective algebraic model widely used in information retrieval (IR). It treats a document or a query as a bag of terms, and uses term weighting schemes such as TF-IDF to weight the terms. Then each document/query is represented as a vector of TF-IDF weights. In particular, term frequency (TF) measures the degree of the relevance between a given document \(d\) and a query term \(t\), which is defined as the number of occurrences of \(t\) in \(d\). Inverse document frequency (IDF) measures the rarity of a term \(t\) in the corpus. In information retrieval, document frequency (DF) for a term \(t\) is the number of documents in the corpus containing \(t\). IDF, the inverse of DF, is the dampened (taking log value) ratio of \(|D|\) (the total number of documents) over DF. Obviously, the weighting issues in ranking-based CF are very similar to those in vector space model.

In this study, we propose VSRank, seeking recommendation accuracy improvement for ranking-based CF through adaptation of the vector space model. Similar (more straightforward) adaptation has been introduced for content-based filtering, demonstrating improvement in recommendation accuracy [Pazzani and Billsus 1997; Zhu et al. 2003; Debnath et al. 2008; Belkin and Croft 1992]. However, this technique has not been investigated in the context of CF.

To adapt the vector space model to ranking-based CF, we consider each user as a document and her pairwise relative preferences as terms. We then use a degree-specialty term weighting scheme resembling TF-IDF to weight the terms. After representing users as vectors of degree-specialty weights, we adopt ranking-based CF techniques to make recommendations for a given user.

Degree-specialty weighting. The key component in the adaptation is the degree-specialty weighting scheme. It is straightforward that relative preferences have different degrees. For example, while both users \(u\) and \(v\) rank item \(i_1\) higher than item \(i_2\), their actual rating scores for \(\{i_1, i_2\}\) may be \(\{5, 1\}\) and \(\{2, 1\}\) respectively, reflecting the fact that user \(u\) prefers \(i_1\) over \(i_2\) much more strongly than user \(v\). Obviously, stronger preferences with larger score differences are more important and should be given a larger degree. Degree resembles term frequency (TF), and a high degree for a preference term from a user can be interpreted in a way that the user repeatedly (frequently) confirms her preference.

Now we explain specialty. A straightforward way of adapting IDF to ranking-based CF is to use the original definition of IDF, which is the dampened ratio of \(|U|\) (the total number of users) over the number of users holding the preference (DF). However, our experiments have returned unsatisfactory results for this method.

A deeper analysis shows that although we conceptually treat preferences as textual terms, they have fundamental differences. While a textual term is un-directional involving only one entity, a preference is directional involving two entities. A preference always has an “enemy”, which is its opposite preference. In light of this, instead of a literal word for word translation, our specialty is essentially a phrasal translation (conveying the sense of the original) of IDF, which measures the rarity of the preference in the users who holding the same or the opposite preferences on the same items.

However, this ratio statistic would suffer from a subtle “small sample” problem because the confidence information (indicated by the number of users) would be cancelled out. We solve this problem by introducing a novel confidence calibration technique that adjusts specialty towards its true value.
While the formal definition will be introduced later, the following example illustrates how specialty truly captures the rarity of preferences.

The effectiveness of VSRank can be understood from another perspective. Terms are features. Feature selection and weighting has been one of the most frequently used techniques in pattern recognition, machine learning and data mining for data analysis, in particular, classification tasks [Han et al. 2011; Bishop 2006]. It eliminates irrelevant, redundant and noisy data. Although numerous classification frameworks and algorithms have been proposed, predicting accuracy is upper bounded by the amount of noise in historical data. Reducing noise has the most direct and immediate effects on predicting accuracy, as it would for recommendation accuracy.

In implementing VSRank, we use two similarity measures, cosine similarity and weighted Kendall tau correlation coefficient, for the discovery of neighborhood users. We also use two preference aggregation methods, an order-based and a score-based, for the prediction of item ranking, resulting in $2 \times 2 = 4$ ranking-based CF algorithms. We conduct extensive experiments on benchmarks in comparison with the state-of-the-art approaches to validate the effectiveness of these algorithms.

**Contribution.** We make the following contributions.

1. We propose VSRank, a framework for adapting the vector space model to ranking-based collaborative filtering for improved recommendation accuracy.
2. We present the novel degree-specialty weighting scheme resembling TF-IDF. We also reveal insightful connections between cosine similarity and Kendall tau rank correlation coefficient.
3. We implement four recommender systems based on VSRank. Extensive experiments on benchmarks demonstrate the promise of our framework.

**Organization.** The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 presents the preliminaries. Section 4 proposes VSRank, the framework for adapting vector space model to ranking-based CF. Section 5 implements four recommender systems based on VSRank. Section 6 reports the experimental results. Section 7 concludes the paper.

2. RELATED WORK

2.1. Recommender Systems

Most existing techniques for recommender systems fall into two categories: collaborative filtering (CF) and content-based filtering. Content-based techniques make recommendations based on regularities in the content information of users and items, where users and items are represented by explicit features [Belkin and Croft 1992; Basu et al. 1998]. CF only utilizes the user-item rating matrix to make predictions and recommendations, avoiding the need of collecting extensive information about items and users. In addition, CF can be easily adopted in different recommender systems without requiring any domain knowledge [Liu and Yang 2008]. Given the effectiveness and convenience, many CF approaches have been proposed, which are either rating-based or ranking-based. Hybrid recommender systems have also been proposed [Burke 2002].

**Rating-based CF.** Rating-based CF techniques can be memory-based or model-based. Memory-based methods make predictions based on similarities between users or items. The user-based paradigm [Resnick et al. 1994; Herlocker et al. 1999] is more common, which estimates the unknown ratings of a target user based on the ratings by a set of neighboring users that tend to rate similarly to the target user. In the item-based paradigm [Deshpande and Karypis 2004; Sarwar et al. 2001], item-item similarity is used to select a set of neighboring items that have been rated by the target user and the ratings on the unrated items are predicted based on his ratings on the neighboring items. Since the number of items is usually much less than the number of users in most applications, item-item similarities are less sensitive to the data sparsity problem. Many commercial systems such as Amazon.com are memory-based since they are relatively easy to implement [Hofmann 2004].
VSRank: A Novel Framework for Ranking-based Collaborative Filtering

Model-based methods estimate or learn a model to make predictions. For example, Vucetic and Obradovic [Vucetic and Obradovic 2005] proposed a regression-based approach to collaborative filtering tasks, which built a collection of simple linear models, and then combined them efficiently to provide rating predictions for an active user. Shani et al. [Shani et al. 2005] used a Markov decision processes (MDPs) model for recommender systems, which viewed the recommendation process as a sequential optimization problem. Si and Jin [Si and Jin 2003] presented a flexible mixture model (FMM) for collaborative filtering. FMM is an extension of partitioning/clustering algorithms, which clusters both users and items together simultaneously without assuming that each user and item should only belong to a single cluster. Tang et al. [Tang et al. 2013] proposed a matrix factorization-based framework LOCABAL, taking advantage of both local and global social context for recommendation. Jiang et al. [Jiang et al. 2012] incorporated social recommendation on the basis of psychology and sociology studies into a probabilistic matrix factorization-based CF algorithm. Comprehensive surveys of rating-based CF can be found in [Adomavicius and Tuzhilin 2005; Gunawardana and Shani 2009; Herlocker et al. 2004; Su and Khoshgoftaar 2009].

**Ranking-based CF.** Ranking-based CF is able to capture the preference similarity between users even if their rating scores differ significantly. Recently, the formulation of recommendation problem is shifting away from rating-based to ranking-based [McNee et al. 2006]. CCF [Yang et al. 2011] learned user preferences using the context of the user behavior of choices in recommender systems, and employed a ranking-oriented latent factor model to characterize the dyadic utility function. CoFiRank [Weimer et al. 2007] used maximum margin matrix factorization to optimize ranking of items for collaborative filtering. EigenRank [Liu and Yang 2008] measured the similarity between users with Kendall tau rank correlation coefficient for neighborhood selection, predicted the relative preferences of items with the preference function, and aggregated these preferences into a total ranking. Liu et al. [Liu et al. 2009] adopted a probabilistic latent preference analysis model pLPA that made ranking predictions by directly modeling user preferences with respect to a set of items rather than the rating scores on individual items. Rendle et al. [Rendle et al. 2009] proposed a Bayesian probabilistic model for personalized ranking from implicit feedback. Cui et al. [Cui et al. 2011] proposed HF-NMF, a hybrid factor non-negative matrix factorization approach for item-level social influence modeling.

It is natural to apply learning to rank to ranking-based recommender systems. In [Shi et al. 2010], ListRank-MF was proposed that combined a list-wise learning to rank algorithm with probabilistic matrix factorization. In [Kahng et al. 2011], a context-aware learning to rank method was proposed that incorporated several context features, such as time and location of users, into the ranking model.

Existing ranking-based CF methods treat relative preferences equally. We adapt the vector space model and weight preferences according to their importance for improved recommendation accuracy. Similar adaptation has been introduced in content-based filtering approaches, demonstrating improvement in recommendation performance [Pazzani and Billsus 1997; Zhu et al. 2003; Debnath et al. 2008; Belkin and Croft 1992]. However, this technique has not been investigated in the context of CF.

### 2.2. Vector Space Model

The vector space model [Baeza-Yates and Ribeiro-Neto 1999] is a standard algebraic model commonly used in information retrieval (IR). It also has other interesting applications. For example, in content-based filtering, the descriptive user profiles can be considered as documents [Pazzani and Billsus 2007] and the vector space model can be applied to make recommendations based on user similarity. In rating-based collaborative filtering, the generalized vector space model can be used to transform vectors of users from the user space into the item space, and then similarity between users and items can be easily measured with cosine similarity [Soboroff and Nicholas 2000]. In image processing, local interest points of images can be clustered, and each cluster can be considered as a visual word [Yang et al. 2007; Kesorn and Poslad 2012], based on which the vector space model can be applied for classification and recognition. In spam detection, features representing
web documents can be partially generated from the vector space model [Niu et al. 2010]. In song sentiment classification, the sentiment vector space model has been proposed to categorize songs into light-hearted and heavy-hearted [Xia et al. 2008], where the song lyrics are regarded as documents and the sentiment words are used to construct the model. In spoken language identification, spoken utterances can be used as term features to build a vector space classifier [Li et al. 2007].

3. PRELIMINARIES

3.1. Collaborative Filtering

The following notations will be used throughout the paper. Let $U$ be a set of users and $I$ be a set of items. In a recommender system, for each user $u \in U$, a set of items $I_u \subseteq I$ are rated by $u$. Let $R$ be a rating matrix, where each element $r_{u,m} \in \mathbb{N}$ is the rating score of the $m^{th}$ item $i_m$ with respect to $u$, and $\mathbb{N}$ is the natural number set indicating different relevance scores.

Collaborating filtering (CF) recommends items to users based on the rating scores predicted by neighborhood users (similar users). In particular, for user $u$, the similarity between $u$ and each user in $U$ is computed from the rating matrix $R$. Then a set of neighborhood users $U_u \subset U$ are selected, based on which recommendations are made.

Ranking-based CF. Ranking-based CF recommends items based on their rankings derived from the rating matrix $R$. The similarity between two users $u$ and $v$, $\tau_{u,v}$, can be computed by the standard Kendall tau rank correlation coefficient [Marden 1995] based on the two rankings from $u$ and $v$ on their common item set.

$$\tau_{u,v} = \frac{N_c - N_d}{\frac{1}{2}N(N-1)},$$  \hspace{1cm} (1)

where $N_c$ and $N_d$ are the numbers of the concordant pairs and discordant pairs respectively.

Let $sgn_{u,v}(m, n)$ be an indicator function such that $sgn_{u,v}(m, n) = 1$ if items $i_m$ and $i_n$ are concordant in $u$ and $v$, and $sgn_{u,v}(m, n) = -1$ if items $i_m$ and $i_n$ are discordant, formally,

$$sgn_{u,v}(m, n) = \begin{cases} 
1, & \text{if } (r_{u,m} - r_{u,n})(r_{v,m} - r_{v,n}) > 0 \\
-1, & \text{if } (r_{u,m} - r_{u,n})(r_{v,m} - r_{v,n}) < 0 
\end{cases},$$  \hspace{1cm} (2)

The sum of $sgn_{u,v}(m, n)$ for all item pairs is $N_c - N_d$, i.e., the number of concordant pairs minus the number of discordant pairs. Thus $\tau_{u,v}$ can be represented as follows:

$$\tau_{u,v} = \frac{1}{\frac{1}{2}N(N-1)} \sum_{m=1}^{N} \sum_{n=m+1}^{N} sgn_{u,v}(m, n)$$  \hspace{1cm} (3)

For user $u$, the preference on a pair of items $(i_m, i_n)$ can be predicted with a preference function $\Psi_u(m, n)$ as follows.

$$\Psi_u(m, n) = \frac{\sum_{v \in U_{m,n}^u} \tau_{u,v} (r_{v,m} - r_{v,n})}{\sum_{v \in U_{m,n}^u} \tau_{u,v}}$$  \hspace{1cm} (4)

where $U_{m,n}^u$ is the set of similar users of $u$ who have rated both items $i_m$ and $i_n$.

Based on the predicted pairwise preferences, a total ranking of items for user $u$ can be obtained by applying a preference aggregation algorithm.
3.2. Vector Space Model

The vector space model [Baeza-Yates and Ribeiro-Neto 1999] is a standard algebraic model commonly used in information retrieval (IR). It treats a textual document as a bag of words, disregarding grammar and even word order. It typically uses TF-IDF (or a variant weighting scheme) to weight the terms. Then each document is represented as a vector of TF-IDF weights. Queries are also considered as documents. Cosine similarity is used to compute similarity between document vectors and the query vector. Large similarity indicate high relevancy of documents with respect to the query.

**TF-IDF.** The term frequency \( TF_{t,d} \) of term \( t \) in document \( d \) is defined as the number of times that \( t \) occurs in \( d \). It positively contributes to the relevance of \( d \) to \( t \).

The inverse document frequency \( IDF_t \) of term \( t \) measures the rarity of \( t \) in a given corpus. If \( t \) is rare, then the documents containing \( t \) are more relevant to \( t \). \( IDF_t \) is obtained by dividing \( N \) by \( DF_t \) and then taking the logarithm of that quotient, where \( N \) is the total number of documents and \( DF_t \) is the document frequency of \( t \), i.e., the number of documents containing \( t \). Formally,

\[
IDF_t = \log_{10} \left( \frac{N}{DF_t} \right)
\]

The TF-IDF value of a term is commonly defined as the product of its \( TF \) and \( IDF \) values.

\[
TF-IDF_{t,d} = TF_{t,d} \times IDF_t.
\]

**Example 3.1.** Let \( d \) be a document containing the term “recommendation” 3 times, then the TF value of “recommendation” for \( d \) is 3. Suppose there are 10 out of \( N = 100 \) documents containing “recommendation”, then the IDF value of “recommendation” is \( \log_{10}(100/10) = 1 \). Then, the TF-IDF value of “recommendation” in \( d \) is \( 3 \times 1 = 3 \).

**Cosine similarity.** Cosine similarity is a standard measure estimating pairwise document similarity in the vector space model. It corresponds to the cosine of the angle between two vectors, and it has the effect of normalizing the length of documents. Let \( q = \langle w_{q,1}, w_{q,2}, \ldots, w_{q,N} \rangle \) and \( d = \langle w_{d,1}, w_{d,2}, \ldots, w_{d,N} \rangle \) be two N-dimensional vectors corresponding a query and a document, their cosine similarity \( s_{q,d} = \frac{q \cdot d}{\|q\| \times \|d\|} \).

4. THE VSRANK FRAMEWORK

In this section, we present VSRank, a novel framework for adapting vector space model to ranking-based collaborative filtering (CF). In VSRank, users are considered as documents and relative preferences are considered as terms. The terms are weighted by a degree-specialty weighting scheme resembling TF-IDF. The target user \( u \) is considered as a query, which is also a document. Then recommendations can be made according to principles of ranking-based CF.

Generally, ranking-based CF works in the following two phases:

— **Phase I: Discovery of neighborhood users.** For each user, Phase I discovers a set of most similar users as the neighborhood users.

— **Phase II: Prediction of item ranking.** Based on the neighborhood users, phase II predicts a ranking list of items by aggregating preferences of neighborhood users for recommendation purpose.

4.1. Representation of Users

We consider users as documents and pairwise relative preferences of items as terms. We adopt a bag of words model, where each user is represented as a bag of relative preferences, instead of a set as in other ranking-based CF methods.

In particular, for a user \( u \), from the set \( I \) of items rated by \( u \), we can derive a set of relative preferences \( \{ i_m \succ i_n | i_m, i_n \in I \cap \forall r_{u,m} > r_{u,n} \} \). Each preference \( i_m \succ i_n \) is considered as a term, and the score difference \( r_{u,m} - r_{u,n} \) indicates the number of “occurrences” of the preference in \( u \).

**Example 4.1.** Suppose user \( u \) has assigned 4, 3, 2 to items \( i_1, i_2 \) and \( i_3 \), respectively. The user \( u \) contains 3 preference terms and can be represented as “\( i_1 \succ i_2, i_1 \succ i_3, i_1 \succ i_3, i_2 \succ i_3 \)”. 

ACM Transactions on Intelligent Systems and Technology, Vol. V, No. N, Article A, Publication date: January YYYY.
4.2. Term Weighting

Degree. Similar to TF, the degree of preference $i_m \succ i_n$ in user $u$ can be defined as the number of occurrences of $i_m \succ i_n$ in $u$. In this paper, we use a logarithm variant of TF. Formally, let $r_{u,m}$ be the rating score of item $i_m$ by user $u$, then the degree of term $i_m \succ i_n$ is defined as:

$$w_{u,i_m \succ i_n}^{(D)} = \log_2 (1 + |r_{u,m} - r_{u,n}|)$$

Specialty. Similar to IDF, we want to use specialty to measure the rarity of preference terms in the set of users. Let $u$ consider preference $i_m \succ i_n$. A straightforward method would be using IDF literally, which is the log value of $\frac{|U|}{N_{i_m \succ i_n}}$, where $|U|$ is the total number of users and $N_{i_m \succ i_n}$ is the DF, that is, the number of users holding the preference $i_m \succ i_n$.

However, we observe that textual terms and preference terms are fundamentally different. While a textual term is un-directional involving only one entity, a preference term is directional involving two entities. A preference term always has an “enemy”, which is its opposite preference term. Also, a textual term is un-directional involving only one entity, a preference term is directional involving two entities. A preference term always has an “enemy”, which is its opposite preference term. Also, a textual term is un-directional involving only one entity, a preference term is directional involving two entities. A preference term always has an “enemy”, which is its opposite preference term.

What is exactly rarity for preference terms? We say that a preference term is rare if there are more opposite preference terms. With the same interpretation, a textual term is rare if there are more documents not containing the term.

The original IDF captures this interpretation of rarity for textual terms, but not for preferences. The nominator of IDF is the total number of documents, which is the number of documents containing the term + the number of documents not containing the term. However, the total number of users is the number of users holding the preference + the number of users holding the opposite preference + the number of users who have not rated both items. Due to the typical sparsity of the rating matrix, most users have not rated both items.

In light of this, instead of using $|U|$ as the nominator, we use “$N_{i_m \succ i_n} + N_{i_m \prec i_n}$” as the nominator. This can be considered as a phrasal translation (conveying the sense of the original) of IDF, instead of a literal word for word one. Example 4.2 provides a clear illustration of this idea.

For each pair of items $(i_m, i_n)$, the relative preferences can be either $i_m \succ i_n$ or $i_m \prec i_n$. For simplicity, we combine the two opposite preference terms into one notation of $i_m \Theta i_n$, where $\Theta \in \{\succ, \prec\}$. Based on the above analysis, a possible specialty would be defined as follows:

$$\lambda_{i_m \Theta i_n} = \log_2 \left( \frac{N_{i_m \succ i_n} + N_{i_m \prec i_n}}{N_{i_m \Theta i_n}} \right)$$

However, this definition would suffer from a subtle “small sample” problem because the ratio in the formula cancels out the confidence information indicated by the number of users. To illustrate the small sample problem, suppose we want to estimate the ratio of the number of males over the number of females in a population. If the sample is too small, the ratio estimate would not be reliable and has a large degree of uncertainty. Note that IDF does not suffer much from this problem because it uses a fixed nominator (total number of documents).

Example 4.2. Let $\{i_1, i_2\}$ be two items. Suppose among the total number of 10,000 users, 1,000 have rated both $i_1$ and $i_2$, where 800 prefer $i_1$ to $i_2$ ($i_1 \succ i_2$) and 200 prefer $i_2$ to $i_1$ ($i_2 \succ i_1$). In this case, $i_2 \succ i_1$ is a rare preference because there are 4 times more users holding the opposite preference. The specialty for preference $i_1 \succ i_2$ is based on $\frac{1000}{800}$ instead of $\frac{1000}{800}$, and the specialty for preference $i_2 \succ i_1$ is based on $\frac{800}{200}$ instead of $\frac{800}{200}$.

Why cannot we use the original IDF? Let $\{i_3, i_4\}$ be two items. Suppose among the total number of 10,000 users, 100 have rated both $i_3$ and $i_4$ with 80 holding preference $i_3 \succ i_4$ and 20 holding preference $i_4 \succ i_3$. In this case, $i_4 \succ i_3$ is a rare preference and $i_3 \succ i_4$ is a popular one. If IDF
is used, then a popular preference \( i_3 \succ i_4 \) would have a much bigger IDF than a rare preference \( i_2 \succ i_1 \) because \( \frac{10000}{80} > \frac{1000}{200} \).

Specialty solves the problem nicely. Without confidence calibration, the two rare preferences would have the same bigger specialty since \( \frac{200}{100} = \frac{100}{20} \), and the two popular preferences would have the same smaller specialty since \( \frac{1000}{200} = \frac{1000}{200} \). With confidence calibration, \( i_2 \succ i_1 \) would have a slightly bigger specialty than \( i_4 \succ i_3 \) and \( i_1 \succ i_2 \) would have a slightly smaller specialty than \( i_3 \succ i_4 \), both due to higher confidence.

To solve this problem, we propose a novel confidence calibration technique. After calibration, the statistic should be brought closer to its true value in the population. The idea is that we define a “prior ratio”, which is the prior knowledge for the ratio. We make an adjustment of the computed statistic towards the prior ratio. When the sample is small, i.e., \( N_{i_m \succ i_n} \) and \( N_{i_m \prec i_n} \) are small, there is more uncertainty and we make a bigger adjustment. Otherwise, we make a small adjustment because we have high confidence in the computed ratio.

Let \(|U|\) be the total number of users. We use the following formula to estimate the “sample size” index for \( i_m \Theta i_n \), which can be used to indicate the confidence level of \( \lambda_{i_m \Theta i_n} \).

\[
\alpha_{i_m,i_n} = \log_2 \left( 1 + (a - 1) \frac{N_{i_m \succ i_n} + N_{i_m \prec i_n}}{|U|} \right)
\]

Finally, the specialty for preference term \( i_m \Theta i_n \) can be defined based on \( \lambda_{i_m \Theta i_n} \) and \( \alpha_{i_m,i_n} \) as follows.

\[
w_{i_m \Theta i_n}^{(S)} = \alpha_{i_m,i_n} \times \lambda_{i_m \Theta i_n} + (1 - \alpha_{i_m,i_n}) \times 1 \tag{7}
\]

In the formula, the default prior specialty is 1, which is the case when \( N_{i_m \succ i_n} = N_{i_m \prec i_n} \) and \( \log_2 \left( \frac{N_{i_m \succ i_n}}{N_{i_m \prec i_n}} \right) = 1 \). \( w_{i_m \Theta i_n}^{(S)} \) drives \( \lambda_{i_m \Theta i_n} \) towards the prior specialty of 1. It makes a bigger adjustment when \( \alpha_{i_m,i_n} \) is small (small confidence) and a smaller adjustment otherwise (large confidence).

**Degree-specialty.** Resembling TF-IDF, degree-specialty is the product of degree and specialty. Specifically, for a user \( u \), the degree-specialty weight of preference term \( i_m \Theta i_n \) is defined using Equations (5) and (7) as follows:

\[
w_{u,i_m \Theta i_n}^{(D)} = w_{u,i_m \Theta i_n}^{(D)} \times w_{i_m \Theta i_n}^{(S)} \tag{8}
\]

**Example 4.3.** Let \( \{i_1, i_2\} \) be two items. Suppose among the total number of 10,000 users, 1,000 have rated both \( i_1 \) and \( i_2 \), where 800 prefer \( i_1 \) to \( i_2 \) (\( i_1 \succ i_2 \)) and 200 prefer \( i_2 \) to \( i_1 \) (\( i_2 \succ i_1 \)). Suppose user \( u \) has assigned scores 2 and 5 and user \( v \) has assigned scores 4 and 3 to items \( i_1 \) and \( i_2 \) respectively.

Then for user \( u \), the degree-specialty for preference term \( i_1 \prec i_2 \) can be computed as follows.

\[
w_{u,i_1 \prec i_2}^{(D)} = \log_2 (1 + |2 - 5|) = 2,
\]

\[
w_{i_1 \prec i_2}^{(S)} = \alpha_{i_1,i_2} \times \lambda_{i_1 \prec i_2} + (1 - \alpha_{i_1,i_2}) \times 1
\]

\[
= \log_{10} 1.9 \times \log_2 \left( \frac{1000}{200} \right) + (1 - \log_{10} 1.9) \times 1
\]

\[
= 1.37,
\]

\[
w_{u,i_1 \prec i_2} \times i_2 = 2 \times 1.37 = 2.74
\]

Similarly, for user \( v \), the degree-specialty for preference term \( i_1 \succ i_2 \) can be computed as follows.
particular, lines 1–5 extract a bag of relative preference terms $T_u$ for each user $u$, forming a set of preference terms $T$. Lines 6–8 compute the specialty weight for each term $t \in T$. Lines 9–14 compute the degree weights, and then obtain a vector of degree-specialty weights for each user.

Lines 15–23 follow a ranking-based CF procedure to make recommendations for each user. In particular, for each user $u$, lines 16–18 compute similarity between $u$ and the rest of users, based on which line 19 selects a set of neighborhood users $U_u$ for $u$. Then lines 21–23 aggregate the preferences of the neighborhood users into a total ranking of items $\tau_u$ for recommendation.

### 4.3. The VSRank Framework

**Pseudocode.** The pseudocode of the VSRank framework is shown in Algorithm 1. Lines 1–14 represent each user as a vector of relative preference terms based on the vector space model. In particular, lines 1–5 extract a bag of relative preference terms $T_u$ for each user $u$, forming a set of preference terms $T$. Lines 6–8 compute the specialty weight for each term $t \in T$. Lines 9–14 compute the degree weights, and then obtain a vector of degree-specialty weights for each user.

Now, let $\{i_3, i_4\}$ be two items. Suppose among the total number of 10,000 users, 100 have rated both $i_3$ and $i_4$ with 80 holding preference $i_3 \succ i_4$ and 20 holding preference $i_4 \succ i_3$. Suppose user $u$ has assigned scores 2 and 5 and user $v$ has assigned scores 4 and 3 to items $i_3$ and $i_4$ respectively. Then for user $u$, the degree-specialty for preference term $i_3 \prec i_4$ can be computed as follows.

\[
\begin{align*}
 w^{(D)}_{u, i_3 \prec i_4} &= \log_2 (1 + |2 - 5|) = 2, \\
 w^{(S)}_{i_3 \prec i_4} &= \alpha_{i_3, i_4} \times \lambda_{i_3 \prec i_4} + (1 - \alpha_{i_3, i_4}) \times 1 \\
 &= \log_{10} 1.09 \times \log_2 \left( \frac{100}{20} \right) + (1 - \log_{10} 1.09) \times 1 \\
 &= 1.05, \\
 w_{u, i_3 \prec i_4} &= 2 \times 1.05 = 2.10
\end{align*}
\]

Similarly, for user $v$, the degree-specialty for preference term $i_3 \succ i_4$ can be computed as follows.

\[
\begin{align*}
 w^{(D)}_{v, i_3 \succ i_4} &= \log_2 (1 + |4 - 3|) = 1, \\
 w^{(S)}_{i_3 \succ i_4} &= \alpha_{i_3, i_4} \times \lambda_{i_3 \succ i_4} + (1 - \alpha_{i_3, i_4}) \times 1 \\
 &= \log_{10} 1.09 \times \log_2 \left( \frac{100}{80} \right) + (1 - \log_{10} 1.09) \times 1 \\
 &= 0.97, \\
 w_{v, i_3 \succ i_4} &= 1 \times 0.97 = 0.97
\end{align*}
\]

From Example 4.3, we can see that strong (high degree) and rare (high specialty) preferences are given higher weights ($2.74 > 0.81$ and $2.10 > 0.97$). We can also see that confidence calibration makes smaller adjustments for specialty with more confidence towards the prior value of 1 ($2.74 > 2.10$ and $0.81 < 0.97$).
ALGORITHM 1: The VSRank Framework.

Input: An item set $I$, a user set $U$, and a rating matrix $R$.
Output: A set of rankings $\{\tau_u\}_{u \in U}$ of items for each user $u \in U$.

1. $T \leftarrow \emptyset$
2. for each $u \in U$ do
3. \hspace{1em} $T_u \leftarrow \text{ExtractTerms}(u, I, R)$
4. \hspace{1em} $T \leftarrow T \cup T_u$
5. end
6. for each $t \in T$ do
7. \hspace{1em} $w_t^{(S)} \leftarrow \text{ComputeSpecialty}(T)$ \hspace{1em} // Eq. 7
8. end
9. for each $u \in U$ do
10. \hspace{1em} for each $t \in T_u$ do
11. \hspace{2em} $w_{u,t}^{(D)} \leftarrow \text{ComputeDegree}(T_u)$
12. \hspace{2em} $w_{u,t} \leftarrow w_t^{(S)} \times w_{u,t}^{(D)}$ \hspace{1em} // Eq. 8
13. \hspace{1em} end
14. end
15. for each $u \in U$ do
16. \hspace{1em} for each $v \in U$ and $u \neq v$ do
17. \hspace{2em} $s_{u,v} \leftarrow \text{ComputeSimilarity}(w_u, w_v)$
18. \hspace{1em} end
19. $U_u \leftarrow \text{SelectNeighbors}\{s_{u,v}\}_{v \in U}$
20. end
21. for each $u \in U$ do
22. \hspace{1em} $\tau_u \leftarrow \text{Aggregate}\{T_v\}_{v \in U_u}$
23. end

Discussion. Let $m$ and $n$ be the numbers of users and items. In VSRank, each user maximally holds $\frac{1}{2}n(n-1)$ preferences. In the worst case, computing degree-specialty weights has a time complexity of $O(mn^2)$. Evaluating similarity between pairs of users has a time complexity of $O(m^2n^2)$. Predicting rankings of items has a time complexity of $O(n^2m^2)$ (see Algorithm 3). In total, VSRank has a time complexity of $O(m^2n^2)$, which is $n$ times higher than that of rating-based CF $O(m^2n)$. Note that the complexity analysis is based on the worst case. First of all, in the real world cases, the rating matrix is very sparse, and each user only rates a very small portion of items [Su and Khoshgoftaar 2009]. Second, as Equations (2) and (3) shown, in ranking-based CF, only pairs of items with different rating scores are considered as preference terms, which further reduces the number of preference terms.

For example, let user $u$ rates $l$ items, where $l \ll n$. Each user can maximal hold $\frac{1}{2}l(l-1)$ preferences, and the time complexity of similarity evaluation in ranking-based CF should be $O(m^2l^2)$ instead. Let the rating scale is from 1 to $s$, and the numbers of items with rating scores of $1, 2, \ldots, s$ are $l_1, l_2, \ldots, l_s$ respectively. The number of preference terms in $u$ is $\frac{1}{2}l(l-1) - \frac{1}{2}\sum l_i(l_i - 1)$.

Furthermore, the recommendation algorithms are performed offline, and can be significantly accelerated via parallel or distributed computing.

5. IMPLEMENTATION OF VSRANK

In implementing VSRank, we use two similarity measures, cosine similarity $\cos_{w,v}$ and Kendall tau correlation coefficient $\tau_{w,v}$, for similarity computation. For preference aggregation, we also use two methods, an order-based and a score-based, to predict item ranking. Depending on the user similarity measure used in Phase I, $\cos_{w,v}$ or $\tau_{w,v}$, and the preference aggregation method used in Phase II, order-based or score-based, we name the resulting algorithms as wVCScore, wVCOder, wTauScore and wTauOrder respectively. Table I shows the four algorithms.
Table I. The four recommender systems in VSRank.

<table>
<thead>
<tr>
<th></th>
<th>Order-based aggregation</th>
<th>Score-based aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \cos_w )</td>
<td>( w^\text{VCO} \text{Order} )</td>
<td>( w^\text{VCO} \text{Score} )</td>
</tr>
<tr>
<td>( \tau^w )</td>
<td>( w^\text{Tau} \text{Order} )</td>
<td>( w^\text{Tau} \text{Score} )</td>
</tr>
</tbody>
</table>

5.1. Similarity Computation

Cosine similarity. The indicator \( p \) of a preference on a pair of items \((i_m, i_n)\) can be defined as a number in \{-1, 1\}, where \( p_{u,(m,n)} = -1 \) for \( i_m \prec i_n \) and \( p_{u,(m,n)} = 1 \) for \( i_m \succ i_n \). Let \( r_{u,m} \) and \( r_{u,n} \) be the rating scores that have been assigned to items \( i_m \) and \( i_n \) respectively by user \( u \). The value for the preference can be written as

\[
p_{u,(m,n)} = \begin{cases} 
-1, & \text{if } r_{u,m} < r_{u,n} \\
1, & \text{if } r_{u,m} > r_{u,n} 
\end{cases}
\]  

(9)

The indicator \( p_{u,(m,n)} \) indicates whether user \( u \) prefers item \( i_m \) to \( i_n \) or vice versa. Remember that we have defined an indicator function \( \text{sgn}_{u,v}(m, n) \) in Equation (2), indicating whether two users \( u \) and \( v \) have a concordant/discordant preference on the pair of items \((i_m, i_n)\). According to Equations (2) and (9), it is easy to prove that

\[
p_{u,(m,n)}p_{v,(m,n)} = \text{sgn}_{u,v}(m, n)
\]  

(10)

With degree-specialty weighting, user \( u \) is represented as a vector of degree-specialty weights \( \hat{w}_u \), where each element is represented as \( \hat{w}_{u,(m,n)} = \Theta_{i_m} \Theta_{i_n} p_{u,(m,n)} \). Then, the similarity between two users \( u \) and \( v \) can be computed by the standard cosine similarity:

\[
\cos_{u,v} = \frac{\hat{w}_u \cdot \hat{w}_v}{\|\hat{w}_u\| \times \|\hat{w}_v\|} = \frac{\sum_{m=1}^{N} \sum_{n=1}^{N} \hat{w}_{u,(m,n)} \times \hat{w}_{v,(m,n)}}{\sqrt{\sum_{m=1}^{N} \sum_{n=1}^{N} \hat{w}^2_{u,(m,n)} \times \sum_{m=1}^{N} \sum_{n=1}^{N} \hat{w}^2_{v,(m,n)}}}
\]  

(11)

Weighted Kendall tau. Shieh [Shieh 1998] proposed \( \tau^w \), a class of weighted variants of Kendall tau rank correlation coefficient that can be used to compute similarity between users \( u \) and \( v \), where each pair of ranks can be weighted separately. Formally,

\[
\tau^w_{u,v} = \frac{\sum_{m=1}^{N} \sum_{n=m+1}^{N} w_{m,n} \times \text{sgn}_{u,v}(m, n)}{\sum_{m=1}^{N} \sum_{n=m+1}^{N} w_{m,n}}
\]  

(12)

where \( w_{m,n} \) is the weight for the pair of items \((i_m, i_n)\), and \( \text{sgn}_{u,v}(m, n) \) is an indicator function, as defined in Equation (2).

The weighted Kendall \( \tau^w \) generalizes the Kendall \( \tau \) rank correlation coefficient, and the latter is a special case when \( w_{m,n} \equiv 1 \), where \( 1 \leq m < n \leq N \).

In estimating similarity between users \( u \) and \( v \), the degree-specialty weight of the item pair \((i_m, i_n)\) in \( \tau^w_{u,v} \) is represented as the product of the weights of \( u \) and \( v \), formally,

\[
w_{m,n} = w_{u,i_m} \Theta_{i_n} w_{v,i_m} \Theta_{i_n}
\]  

(13)
Thus the weighted Kendall tau correlation coefficient can be rewritten as follows:

\[
\tau_{w}^{u,v} = \frac{\sum_{m=1}^{N} \sum_{n=m+1}^{N} w_{u,i_{m}} \Theta_{i_{n}} w_{v,i_{m}} \Theta_{i_{n}} sgn_{u,v}(m, n)}{\sum_{m=1}^{N} \sum_{n=m+1}^{N} w_{u,i_{m}} \Theta_{i_{n}} w_{v,i_{m}} \Theta_{i_{n}}}
\]

(14)

Relationship between \(\cos_{u,v}\) and weighted \(\tau_{w}^{u,v}\). Now we reveal interesting connections between cosine similarity \(\cos_{u,v}\) and weighted Kendall tau rank correlation coefficient \(\tau_{w}^{u,v}\).

**Theorem 5.1.** \(\cos_{u,v} = \tau_{w}^{u,v} \cos(w_{u}, w_{v})\).

**Proof.**

\[
\hat{\mu}_{u} \cdot \hat{\mu}_{v} = \sum_{m=1}^{N} \sum_{n=m+1}^{N} (w_{u,i_{m}} \Theta_{i_{n}} p_{u,(m,n)}) (w_{v,i_{m}} \Theta_{i_{n}} p_{v,(m,n)})
\]

According to Equations (10)

\[
\hat{\mu}_{u} \cdot \hat{\mu}_{v} = \sum_{m=1}^{N} \sum_{n=m+1}^{N} (w_{u,i_{m}} \Theta_{i_{n}} w_{v,i_{m}} \Theta_{i_{n}} sgn_{u,v}(m, n))
\]

Since

\[
\|\hat{\mu}_{u}\| \times \|\hat{\mu}_{v}\| = \sqrt{\sum_{m=1}^{N} \sum_{n=m+1}^{N} w_{u,i_{m}}^{2} \Theta_{i_{n}}} \sqrt{\sum_{m=1}^{N} \sum_{n=m+1}^{N} w_{v,i_{m}}^{2} \Theta_{i_{n}}}
\]

\[
= \sqrt{\sum_{m=1}^{N} \sum_{n=m+1}^{N} w_{u,i_{m}}^{2} \Theta_{i_{n}}} \sqrt{\sum_{m=1}^{N} \sum_{n=m+1}^{N} w_{v,i_{m}}^{2} \Theta_{i_{n}}}
\]

\[
= \|w_{u}\| \times \|w_{v}\| = \frac{w_{u} \cdot w_{v}}{\cos(w_{u}, w_{v})}
\]

Hence
\[
\cos_{u,v} = \frac{\hat{w}_u \cdot \hat{w}_v}{||\hat{w}_u|| \times ||\hat{w}_v||}
\]
\[
= \frac{1}{N \sum_{m=1}^{N} \sum_{n=m+1}^{N} (w_{u,i_m} \Theta_{i_n} w_{v,i_m} \Theta_{i_n} sgn_{u,v}(m, n)) \cos(w_u, w_v)}
\]
\[
= \tau_{u,v} \cos(w_u, w_v)
\]

\[\square\]

**Corollary 5.2.** Without weighting, cosine similarity \(\cos_{u,v}\) is equivalent to the Kendall tau rank correlation coefficient \(\tau_{u,v}\).

The corollary can be easily derived from Theorem 5.1 for the special unweighted case of \(w_u = w_v = \langle 1, 1, \ldots, 1 \rangle\).

**Discussion.** The above theoretical results reveal valuable insights. Existing ranking-based CF methods do not weight preferences, and Kendall tau rank correlation coefficient \(\tau_{u,v}\) is the standard similarity measure. We have showed that it is equivalent to cosine similarity for the unweighted case after adapting the vector space model. However, with preference weighting, \(\cos_{u,v} = \tau_{u,v} \cos(w_u, w_v)\). Then comparing \(\cos_{u,v}\) with \(\tau_{u,v}(w_u, w_v)\), the former incorporates length normalization whereas the latter does not. This explains our experimental results that the former performed better than the latter.

### 5.2. Ranking Prediction

As introduced in Section 4, ranking-based CF discovers a set of most similar users in Phase I and predicts a ranking list of items for recommendation in Phase II.

We have discussed Phase I in Section 5.1. In this section, we discuss Phase II, ranking prediction, where we aggregate the partial preference rankings of the neighborhood users into a total ranking of items that can be used for recommendation. Cohen et al. [Cohen et al. 1999] proved that it is an NP-hard problem.

Aggregation algorithms can be classified into two categories: order-based and score-based [Aslam and Montague 2001; Gleich and Lim 2011]. The former only uses relative preference orders of items to generate a total ranking of items with an order-based aggregation function. The latter predicts a score for each pair of items to indicate the preference degree of items, based on which a total ranking of items are generated with a score-based aggregation function. Generally speaking, order-based algorithms are less biased, whereas score-based algorithms are more effective for sparse data [Baskin and Krishnamurthi 2009].

In this study, we provide two algorithms for preference aggregation. First of all, we adopt Schulze’s method for order-based aggregation. The Schulze method [Schulze 2003] is a voting system to create a sorted list of winners with votes iteratively, which satisfies the properties of pareto, monotonicity, resolvability, independence of clones, and reversal symmetry. For score-based aggregation, we use a greedy method, which is straightforward, easy to implement, yet highly effective.

**Order-based preference aggregation.** For a given user \(u\), order-based preference aggregation attempts to maximize the number of consistences between the aggregated ranking \(\tau_u\) and preference rankings from the neighborhood users, and simultaneously minimize the number of inconsistences.
Let $U_{u}^{m,n}$ be the set of neighborhood users of $u$ who have rated both items $i_m$ and $i_n$, the objective function for optimization is given as follows:

$$\arg \max \sum_{\forall (i_m,i_n): \tau_u(i_m)<\tau_u(i_n)} \sum_{v \in U_{u}^{m,n}} p_{v,(m,n)}$$

(15)

where $p_{v,(m,n)}$ indicates whether user $v$ prefers $i_m$ to $i_n$ or vice versa, outputting 1 or $-1$ respectively (see Equation (9)), and $\tau_u(i_m) < \tau_u(i_n)$ indicates that $i_m$ is prior to $i_n$ in the aggregated ranking $\tau_u$.

In this study, we adopt Schulze method to implement order-based aggregation as shown in Algorithm 2. The algorithm has a time complexity of $O(n^3)$, where $n$ is the number of items.

**ALGORITHM 2:** The Schulze method for order-based preference aggregation.

**Input**: An item set $I$, a user $u$, and a preference prediction function $\Psi_u$.

**Output**: A ranking $\tau_u$ of items for user $u$.

```latex
\begin{align*}
&M_{I \times I} \leftarrow 0 \\
&\text{for } m \leftarrow 1 \text{ to } |I| \text{ do} \\
&\text{\quad for } n \leftarrow 1 \text{ to } |I| \text{ and } m \neq n \text{ do} \\
&\text{\quad\quad } M_{m,n} \leftarrow N_{i_m > i_n} \\
&\text{\quad end} \\
&\text{end} \\
&\text{for } m \leftarrow 1 \text{ to } |I| \text{ do} \\
&\text{\quad for } n \leftarrow 1 \text{ to } |I| \text{ and } m \neq n \text{ do} \\
&\text{\quad\quad } M_{n,k} \leftarrow \max(M_{n,k}, \min(M_{n,m}, M_{m,k})) \\
&\text{\quad end} \\
&\text{end} \\
&\text{for each } i_m \in I \text{ do} \\
&\text{\quad } \tau_u(i_m) \leftarrow \sum_{\forall i_n \in I \backslash \{i_m\}} 1(M_{m,n} > M_{n,m}) \\
&\text{end}
\end{align*}
```

In the algorithm, line 1 introduces an $|I| \times |I|$ matrix $M$, where each element $M_{m,n}$ indicates the relative preference degree of the pair of items $(i_m, i_n)$ for the neighborhood users of $u$. Then lines 2–5 initialize each element $M_{m,n}$ of the matrix as the number of preferences $i_m > i_n$ voted by the neighborhood users.

Lines 7–12 estimate the highest preference degree for each element $M_{m,n}$ of the matrix iteratively, and update it with $\max(M_{n,k}, \min(M_{n,m}, M_{m,k}))$. For example, suppose that currently there are 10 neighborhood users who prefer $i_n$ to $i_k$, 20 neighbors who prefer $i_n$ to $i_m$, and 15 neighbors who prefer $i_m$ to $i_k$. There are two preference paths from $i_n$ to $i_k$: one from $i_n$ to $i_k$ directly with the preference degree of 10, and the other from $i_n$ through $i_m$ to $i_k$ with the preference degree of $\min(20, 15) = 15$. Thus the preference path with the highest preference degree is the latter with a value of $\max(10, \min(20, 15)) = 15$.

Lines 14–16 iteratively produce a total ranking of items. In particular, the item $i_m$ is prior to $i_n$ if $M_{m,n} > M_{n,m}$, and the rank position of $i_m$ is equal to the number of occurrences of $M_{m,n} > M_{n,m}$ for any other item $i_n$.

**Score-based preference aggregation.** Order-based algorithms only use the relative preference orders of items to generate a total ranking. On the other hand, score-based algorithms consider the similarity scores between the given user and her neighborhood users as weights of their preferences. Let $s_{u,v}$ be the similarity between two users $u$ and $v$. For a given user $u$, score-based preference
aggregation attempts to maximize the weighted consistences between the aggregated ranking \( \tau_u \) and the preferences of the neighborhood users of \( u \):

\[
\arg \max \sum_{v \in U_{m,n}^u} s_{u,v} p_{v,(m,n)},
\]

(16)

Similar to EigenRank [Liu and Yang 2008], we define a preference prediction function \( \Psi_u(i_m, i_n) \) as follows:

\[
\Psi_u(i_m, i_n) = \sum_{v \in U_{m,n}^u} s_{u,v} p_{v,(m,n)} \sum_{v \in U_{m,n}^u} s_{u,v} \]  

(17)

For a given user \( u \), the preference prediction function \( \Psi_u(i_m, i_n) : I \times I \rightarrow \mathbb{R} \) assigns real number confidence scores to preferences, where \( I \) is the item set and \( \mathbb{R} \) is the real number set. \( \Psi_u(i_m, i_n) > 0 \) indicates that item \( m \) is more preferable to \( n \) by user \( u \) and vice versa. The magnitude of the preference function \( |\Psi_u(i_m, i_n)| \) implies the evidence of the preference, and a value of zero means that there is no preference between the two items. Thus the objective function for optimization in score-based approach can be rewritten as follows:

\[
\arg \max \sum_{v \in U_{m,n}^u} \Psi_u(i_m, i_n)
\]

(18)

In this study, we provide a greedy method for score-based preference aggregation as shown in Algorithm 3.

**ALGORITHM 3:** The greedy method for score-based preference aggregation.

**Input:** An item set \( I \), a user \( u \), and a preference prediction function \( \Psi_u \)

**Output:** A ranking \( \tau_u \) of items for user \( u \)

1. \( N \leftarrow |I| \)
2. for each \( i \in I \) do
3. \( \pi_u(i) \leftarrow \sum_{j \in I} \Psi_u(i, j) - \sum_{j \in I} \Psi_u(j, i) \)
4. end
5. while \( I \neq \emptyset \) do
6. \( t \leftarrow \arg \max \pi_u(i) \)
7. \( \tau_u(t) \leftarrow N - |I| \)
8. \( I \leftarrow I \setminus \{t\} \)
9. for each \( i \in I \) do
10. \( \pi_u(i) \leftarrow \pi_u(i) + \Psi_u(t, i) - \Psi_u(i, t) \)
11. end
12. end

For a given user \( u \), Algorithm 3 assigns to each item \( i \in I \) a potential value \( \pi_u(i) \), which is the sum of the evidence scores of the preferences starting with \( i \) minus the sum of the evidence scores of the preferences ending with \( i \) (lines 2-4). Then the algorithm iteratively produces the rank of each item in a greedy strategy until \( I \) is empty (lines 5-12). In particular, the algorithm first picks some
item $i$ with maximum potential value, and assigns it a rank $\tau_u(i) = N - |I|$ (lines 6-7). Then it deletes $i$ from $I$ (line 8), and updates the potential values of the remaining items (lines 9-11).

The algorithm has a time complexity of $O(n^2)$, where $n$ is the number of the items. It can be proved to have an approximation ratio of 2, i.e., $\Lambda_S(\tau_u) \geq \frac{1}{2} \Lambda_S(\tau_u^*)$ [Cohen et al. 1999].

6. EXPERIMENTS

6.1. Methodology

Datasets. We used two real movie rating datasets in our experiments, EachMovie and MovieLens. The EachMovie dataset contains about 2.8 million ratings, which are made by 74,418 users on 1648 movies. The MovieLens dataset consists of 1 million ratings assigned by 6040 users to a collection of 3952 movies. The EachMovie rating scale is from 0 to 5, while the MovieLens rating scale is from 1 to 5. Table II lists the detailed statistics about the two datasets.

<table>
<thead>
<tr>
<th></th>
<th>EachMovie</th>
<th>MovieLens</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of users</td>
<td>74,418</td>
<td>6,040</td>
</tr>
<tr>
<td>Number of movies</td>
<td>1,648</td>
<td>3,952</td>
</tr>
<tr>
<td>Number of ratings</td>
<td>2,811,983</td>
<td>1,000,209</td>
</tr>
<tr>
<td>Rating scales</td>
<td>0–5</td>
<td>1–5</td>
</tr>
</tbody>
</table>

Evaluation measures. For rating-based collaborative filtering, the standard evaluation criterion is the rating prediction accuracy. Commonly used accuracy measures include the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE). Both measures depend on difference between true rating and predicted rating. Since our study focuses on improving item rankings instead of rating prediction, we employ two ranking-oriented evaluation measures: Normalized Discounted Cumulative Gain (NDCG) [Järvelin and Kekäläinen 2002] and Mean Average Precision (MAP). They are popular in information retrieval for evaluating ranked results, where documents are assigned graded relevance judgements in NDCG and binary relevance judgements in MAP.

In the context of collaborative filtering, item ratings assigned by users can naturally serve as relevance judgements. The NDCG metric is evaluated over some number $n$ of the top items on the ranked item list. Let $U$ be the set of users and $r_{u,p}$ be the rating score assigned by user $u$ to the item at the $p$th position of the ranked list from $u$. The NDCG at the $n$th position with respect to the given user $u$ is defined as follows.

$$\text{NDCG}_u@n = \sum_{p=1}^{n} \frac{2^{r_{u,p}} - 1}{\log(1 + p)}$$  (19)

NDCG at the $n$th position takes the mean of the NDCG values at the same position over the set of users $U$.

$P@n$ represents the precision within the top $n$ results of the ranked list of items for a user. Average precision (AP) for user $u$ is defined as the average of the $P@n$ values for all relevant items:

$$\text{AP}_u = \frac{\sum_{p=1}^{N} (P@n \times \text{rel}_{u,p})}{\# \text{ relevant items for user } u},$$  (20)

7http://www.grouplens.org/node/12
where \( rel(n) \) is a binary function mapping a document to either 1 (relevant) or 0 (irrelevant). In this experiment, we regarded the rating scores of 5 as relevant while scores less than 5 as irrelevant. MAP takes the mean of the AP values over the set of users \( U \).

**Comparison partners.** We used three state-of-the-art ranking-based collaborative filtering algorithms, EigenRank [Liu and Yang 2008], CCF [Yang et al. 2011] and CoFiRank [Weimer et al. 2007], as our main comparison partners. In [Yang et al. 2011], two CCF algorithms of CCF-Softmax and CCF-Hinge were provided with softmax and hinge loss functions respectively, achieving similar recommendation performances. In our experiments, we used CCF-Hinge for comparison. In addition, we also included comparisons with UVS [John S. Breese 1998], a conventional user-based collaborative filtering method. UVS measured similarity between users using the vector cosine similarity, and then ranked the items for each user according to their predicted rating scores for the purpose of obtaining a ranking of items.

**Experimental setup.** In our experiments, we randomly selected 80% rated items for training and used the remaining 20% for testing. In order to guarantee that there are adequate number of common rating items between each neighborhood user and the target user, we filtered those users who have rated less than 50 items in MovieLens and 100 items in EachMovie. We ran each algorithm 5 times and reported the average performance.

### 6.2. Accuracy

In the first series of experiments, we evaluated the accuracy performance of wVOrder, wTauOrder, wVScore and wTauScore, in comparison with EigenRank, CoFiRank, CCF-Hinge and UVS on EachMovie and MovieLens.

Figures 1(a) and 1(b), and Table III show the comparison of performance evaluated with NDCG and MAP measures. From the figures and the table we can see that:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>wVOrder</th>
<th>wTauOrder</th>
<th>wVScore</th>
<th>wTauScore</th>
<th>EigenRank</th>
<th>CCF-Hinge</th>
<th>CoFiRank</th>
<th>UVS</th>
</tr>
</thead>
<tbody>
<tr>
<td>EachMovie</td>
<td>0.5284</td>
<td>0.5184</td>
<td>0.5267</td>
<td>0.5201</td>
<td>0.5036</td>
<td>0.5224</td>
<td>0.5032</td>
<td>0.4789</td>
</tr>
<tr>
<td>MovieLens</td>
<td>0.5806</td>
<td>0.5791</td>
<td>0.5808</td>
<td>0.5808</td>
<td>0.5731</td>
<td>0.5743</td>
<td>0.5759</td>
<td>0.5189</td>
</tr>
</tbody>
</table>
(1) Our proposed degree-specialty weighting scheme can discover a more accurate set of neighborhood users, resulting in improved recommendation accuracy. For the two benchmark datasets, the four recommender systems wVCScore, wTauOrder, wVCScore and wTauOrder outperformed all other comparison partners.

(2) Cosine similarity \( \text{cos}^w \) used in our vector space model is more effective than the weighted Kendall tau rank correlation coefficient \( \tau^w \), evidenced by the fact that wVCScore and wVCOrder outperformed wTauScore and wTauOrder respectively.

(3) Order-based aggregation is less variance than score-based, evidenced by the fact that the standard deviations of wVCOrder and wTauOrder is much smaller than those of wVCScore and wTauScore. For example, for EachMovie, the standard deviations of wVCOrder and wTauOrder are 0.0007 and 0.0006 on NDCG@1–2 comparing to 0.0045 and 0.0040 for those of wVCScore and wTauScore. For MovieLens, the standard deviations of wVCOrder and wTauOrder are 0.0006 and 0.0021 on NDCG@1–2 comparing to 0.0097 and 0.0040 for those of wVCScore and wTauScore.

(4) Ranking-based collaborative filtering can be advantages over rating-based methods on the NDCG evaluation measure. In our experiments, all the ranking-based methods outperformed the rating-based method UVS. For example, for EachMovie, wVCScore achieved 0.7402 and 0.7504 on NDCG@1–2 comparing to 0.6471 and 0.6609 for UVS, gaining 14.39% and 13.54% improvement respectively. For MovieLens, wVCScore achieved 0.7467 and 0.7489 on NDCG@1–2 comparing to 0.6723 and 0.688 for UVS, gaining 11.07% and 8.85% improvement respectively.

(5) From Table III we can see that, while in general our algorithms outperformed the comparison partners in MAP, their advantages were not as obvious as in NDCG. For example, CCF-Hinge outperformed two of our four algorithms wTauOrder and wTauScore on EachMovie. This is mainly because unlike NDCG, MAP is not the most appropriate measure for multi-level ratings. MAP natively handles binary ratings, whereas EachMovie contains 6-level ratings (0–5) and MovieLens contains 5-level ratings (1–5).

### 6.3. Sensitivity of Neighborhood Size

The size of the neighborhood has significant impact on the prediction quality for conventional collaborative filtering [Herlocker et al. 1999; Liu and Yang 2008]. For example, for EigenRank, the NDCG values gradually increase as the neighborhood size increases, and reach the peaks at the neighborhood size of 100 [Liu and Yang 2008].
We studied the sensitivity of this parameter on our methods. We conducted a series of experiments on wVCOrder and wVCScore for the EachMovie and MovieLens datasets with the number of neighborhood users varying from 1 to 100. The experiment results are reported in Figures 2 and 3. From the results, we can see that the curves tend to be flat for EachMovie, and even decline for MovieLens after the neighborhood size exceeds 20. Based on the results, we have the following observations.

(1) The size of the neighborhood has an impact on the prediction quality of our methods. Prediction is accurate when the neighbors are very similar to the target user. When the neighborhood size exceeds 20, the performance starts to decrease because more dissimilar users are selected into the neighborhood, introducing noise to $\Psi(m, n)$.

(2) Effective term weighting scheme can significantly benefit discovery of good neighborhood. wVCOrder and wVCScore are able to discover the most similar users with the neighborhood size of 20, comparing to 100 for EigenRank, the conventional ranking-based collaborative filtering method.
6.4. Discussions on Specialty

Based on the experiments, we have the following observations: (1) Specialty is more appropriate than the original IDF for measuring rarity of preference terms in the context of ranking-based collaborative filtering. (2) Confidence calibration can be used to adjust specialty towards its true value.

In particular, we applied wVCScore, which uses specialty, on the MovieLens dataset. We then repeated the experiments with two modified recommender systems of wVCScore-IDF and wVCScore-NC. The former replaces specialty with the original IDF, and the latter removes confidence calibration from specialty. The comparison results are reported in Figure 4, from which we can see that:

(1) “Specialty” significantly outperformed “IDF”. For example, wVCScore achieved 0.7795 and 0.7989 on NDCG@4–5 comparing to 0.7192 and 0.7228 for wVCScore-IDF, gaining 8.4% and 10.5% improvement respectively. With “Degree-IDF”, the performance of wVCScore-IDF is even slightly worse than that of EigenRank, the ranking-based CF without weighting.

(2) “Confidence calibration” can help improve accuracy. For example, wVCScore achieved 0.7467 and 0.7989 on NDCG@1 and 5 comparing to 0.7218 and 0.7802 for wVCScore-NC, gaining 3.4% and 2.4% improvement respectively.

7. CONCLUSION

In this paper, we have proposed VSRank, a framework for adapting the vector space model to ranking-based collaborative filtering for improved recommendation accuracy. Different from existing ranking-based CF methods that treat each user as a set of preferences, we adopt the bag of words model capturing the “frequency” of preferences. Different from existing ranking-based CF methods that treat preferences equally, we use a novel degree-specialty weighting scheme resembling TF-IDF. Users are represented as vectors of degree-specialty weights and ranking-based CF techniques are used to predict a ranking of items for accurate recommendation to the target user. Comprehensive experiments have validated the effectiveness of our framework.

There are several interesting directions for future work. Firstly, other ranking-based similarity measures can be experimented for improving neighborhood quality. Secondly, knowing that there are many TF-IDF variants, we plan to investigate other possible variants of degree-specialty and study their performance in different applications. Last but not least, the proposed adaptation framework is not limited to ranking-based CF. We plan to explore a similar adaptation of the vector space model to rating-based CF and examine its effectiveness.

Acknowledgement

This work was supported in part by the National Science Foundation (OCI-1062439 and CNS-1058724), the Natural Science Foundation of China (61272240 and 71171122), the Humanity and Social Science Foundation of Ministry of Education of China (12YJC630211), the Specialized Research Foundation of Ministry of Education of China for Returned Oversea Scholars, the Natural Science Foundation of Shandong Province of China (2012BSB01550), and the Specialized Research Foundation of Jinan for Returned Oversea Scholars (20120201).

REFERENCES


