CLiMF: Learning to Maximize Reciprocal Rank with Collaborative Less-is-More Filtering

Yue Shi*
Delft University of Technology
y.shi@tudelft.nl

Alexandros Karatzoglou
Telefonica Research
alexk@tid.es

Linas Baltrunas
Telefonica Research
linas@tid.es

Martha Larson
Delft University of Technology
m.a.larson@tudelft.nl

Nuria Oliver
Telefonica Research
nuriao@tid.es

Alan Hanjalic
Delft University of Technology
a.hanjalic@tudelft.nl

ABSTRACT

In this paper we tackle the problem of recommendation in the scenarios with binary relevance data, when only a few (k) items are recommended to individual users. Past work on Collaborative Filtering (CF) has either not addressed the ranking problem for binary relevance datasets, or not specifically focused on improving top-k recommendations. To solve the problem we propose a new CF approach, *Collaborative Less-is-More Filtering (CLiMF)*. In *CLiMF* the model parameters are learned by directly maximizing the Mean Reciprocal Rank (MRR), which is a well-known information retrieval metric for capturing the performance of top-k recommendations. We achieve linear computational complexity by introducing a lower bound of the smoothed reciprocal rank metric. Experiments on two social network datasets demonstrate the effectiveness and the scalability of *CLiMF*, and show that *CLiMF* significantly outperforms a naive baseline and two state-of-the-art CF methods.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering*

General Terms

Algorithms, Experimentation, Performance

Keywords

Collaborative filtering, learning to rank, less is more, matrix factorization, mean reciprocal rank

*Part of this work was conducted when the first author was an intern at Telefonica Research, Barcelona.

RecSys'12, September 9–13, 2012, Dublin, Ireland.

1. INTRODUCTION

Collaborative Filtering (CF) [1] methods are at the core of most recommendation engines in online web-stores and social networks. The main underlying idea behind CF methods is that users that shared common interests in the past would still prefer similar products/items in the future [22]. While a lot of the CF literature has been devoted to recommendation scenarios where explicit user feedback is present (i.e., typically ratings), CF has also shown to be very valuable in scenarios with only implicit feedback data [12], e.g., the counts of a user watching a TV show, the counts of a user listening to songs of an artist, etc. These counts can be interpreted as a measure of preference and thus a proxy to explicit feedback.

However, in some scenarios even the "count" information is not available, while only binary relevance data exists, e.g., the friendship between users in a Online Social Network, the follow relationship between users (or between a user and an event, etc.) in Twitter[1] or the dating history in online dating sites [20], etc. Specifically, in these scenarios, we use "1" for a given user-item pair to denote that the user has an interaction (e.g., friendship, follow) with the item, and "0" otherwise. Typically the observed interactions are regarded as positive signals (i.e., indicating relevant items), and although not all items without observed interactions are irrelevant it is safe to assume the vast majority of these items will be irrelevant for the user. In other words, for a given user, the signal "0" indicates an item set containing unobserved items that could be relevant but are most likely irrelevant. One of the most typical CF methods for those scenarios is item-based CF [9, 15], in which an item-item similarity matrix is first computed, and users are recommended items that are most similar to their past relevant items. However, item-based CF approaches typically require expensive computations in order to construct the similarity matrix. They are thus not a sound solution for large scale scenarios.

Bayesian Personalized Ranking (BPR) [21] has been recently proposed as a state-of-the-art recommendation algorithm for situations with binary relevance data. The optimization criterion of BPR is essentially based on pair-wise comparisons between relevant and a sample of irrelevant items. This criterion leads to the optimization of the Area Under the Curve (AUC). However, the AUC measure does not reflect well the quality of the recommendation lists, since

[1]http://twitter.com/

it is not a top-biased measure [33], i.e., the position at which the pairwise comparisons are made is irrelevant to the contribution to the loss: mistakes at the lower ranked positions are penalized as equally as mistakes in higher ranked positions, which is not the desired behavior in a ranked list.

In view of the drawbacks of previous work, we propose a new CF approach, *Collaborative Less-is More Filtering (CLiMF)*, that is tailored to recommendation domains where only binary relevance data is available. *CLiMF* models the data by means of directly optimizing the Mean Reciprocal Rank (MRR) [28], a well-known evaluation metric in Information Retrieval (IR). Given the analogy between query-document search and user-item recommendation, we can define the Reciprocal Rank (RR) for a given recommendation list of a user, by measuring how early in the list (i.e. how highly ranked) is the first relevant recommended item. The MRR is the average of the RR across all the recommendation lists for individual users. MRR is a particularly important measure of recommendation quality for domains that usually provide users with only few but valuable recommendations (i.e., the *less-is-more* effect [7]), such as friends recommendation in social networks where top-3 or top-5 performance is important.

Taking insights from the area of learning to rank and integrating latent factor models from CF, *CLiMF* directly optimizes a lower bound of the smoothed RR for learning the model parameters, i.e., latent factors of users and items, which are then used to generate item recommendations for individual users.

Our contributions in this paper can be summarized as:

- We present a new CF approach, *CLiMF*, for MRR optimization for scenarios with binary relevance data. We demonstrate that *CLiMF* outperforms other state-of-the-art approaches with respect to making only a few but relevant recommendations.
- We introduce a lower bound of the smoothed RR measure, significantly reducing the computational complexity of RR optimization, and enabling *CLiMF* to scale for large datasets.

The paper is organized as follows. In Section 2 we discuss the related work and position our paper with respect to it. Section 3 presents in detail the proposed *CLiMF* model. Our experimental evaluation is described in Section 4, followed by a summary and conclusions in Section 5.

2. RELATED WORK

The work presented in this paper closely relates to the research on ranking-oriented CF and learning to rank. In the following, we briefly review related work.

2.1 Ranking-oriented CF

A large portion of the Recommender Systems literature has been devoted to the rating prediction problem, as defined in the Netflix prize competition[2]. Latent factor models and in particular Matrix Factorization (MF) techniques, have been shown to be particularly effective [2, 13, 23] for this problem. The main idea underlying MF is to extract latent factor U_i, V_j vectors for each user and item in the dataset so that the inner product of these factors $f_{ij} = \langle U_i, V_j \rangle$ fits the observed ratings.

[2]http://www.netflixprize.com/

Several state-of-the-art ranking-oriented CF, that extend upon MF techniques, have been recently proposed. These approaches typically use a ranking oriented objective function to learn the latent factors of users and items, e.g., CofiRank [30], collaborative competitive filtering (CCF) [32], and OrdRec [14]. The *CLiMF* model presented in this paper can also be regarded as an extension to conventional MF, while it introduces several new characteristics that are presented in Section 3.4, compared to the state-of-the-art.

A ranking-oriented CF that extends memory-based (or similarity-based) approaches has been proposed in EigenRank [16]. Moreover extensions to probabilistic latent semantic analysis [11] that optimize a ranking objective have been proposed in pLPA [17]. However, these methods are all designed for recommendation scenarios with explicit graded relevance scores from users to items.

For the use scenarios with only implicit feedback data, one of the first model-based methods was introduced in [12], where an extension of MF is proposed by weighting each factorization of user-item interaction proportionately to the count of the interactions. A similar approach, one-class collaborative filtering [19], was also proposed to exploit weighting schemes for the factorizations of missing data, which are taken as non-positive examples. However, the computational cost of that work could be inflated due to the large number of non-positive data. In this paper, we study the problem of generating recommendations for the scenarios with only binary relevance data, i.e., where even the count of user-item interaction is not available. In addition, our work directly takes into account an evaluation metric, MRR, when developing the recommendation model, which is also substantially different from the work of [12, 19].

The most similar work to ours is Bayesian personalized ranking (BPR) [21], since it also optimizes a ranking loss (AUC) and deals with binary relevance data. The main benefits of using *CLiMF* lies in its performance in terms of top-k recommendations (i.e the fraction of relevant items at the top k positions of the list), an issue not addressed by the BPR model. Note that we also leave the detailed discussion of the relationship between *CLiMF* and BPR to Section 3.4, after the presentation of the *CLiMF* model.

2.2 Learning to Rank

Learning to Rank (LTR) has been an active research topic in Machine Learning, Information Retrieval [18] and Recommender Systems [3, 24, 30]. The work in this paper is closely related to one branch of LTR that focuses on direct optimization of IR metrics, for which the main difficulty lies in their non-smoothness with respect to the predicted relevance scores [4]. The approaches proposed in this branch of LTR approximate the optimization of IR measures either by minimizing convex upper bounds of loss functions that are based on the evaluation measures [5, 30, 31], e.g., SVM^{MAP} [33], or by optimizing a smoothed version of an evaluation measure, e.g., $SoftRank$ [27] and *generalized* $SoftRank$ [6].

In this paper, we also propose to approximate the Mean Reciprocal Rank (MRR) with a smoothed function. However, our work is different from aforementioned work not only in that we target the application scenario of recommendation rather than query-document search, but also in that we propose an algorithm (CLiMF) that makes the optimization of the smoothed MRR tractable and scalable. We also

provide insights about the ability of *CLiMF* to recommend relevant items in the top positions of a recommendation list.

3. CLIMF

In this section, we present the *CLiMF*, Collaborative Less-is-More Filtering, algorithm. We first introduce a smoothed version of Reciprocal Rank by taking insights from the area of learning to rank. Then, we derive a lower bound of the smoothed reciprocal rank, and formulate an objective function for which standard optimization methods can be deployed. Finally, we discuss the characteristics of the proposed *CLiMF* model and its relation to other state-of-the-art recommendation models.

3.1 Smoothing the Reciprocal Rank

The definition of reciprocal rank of a ranked list for user i, as defined in information retrieval [28], can be expressed as:

$$RR_i = \sum_{j=1}^{N} \frac{Y_{ij}}{R_{ij}} \prod_{k=1}^{N} (1 - Y_{ik} \mathbb{I}(R_{ik} < R_{ij})) \tag{1}$$

in which N is the number of items, Y_{ij} denotes the binary relevance score of item j to user i, i.e., $Y_{ij} = 1$ if item j is relevant to user i, 0 otherwise. $\mathbb{I}(x)$ is an indicator function that is equal to 1, if x is true, otherwise 0. R_{ij} denotes the rank of item j in the ranked list of items for user i. Note that the items are ranked in a descending order according to their predicted relevance scores for user i. Clearly, RR_i is dependent on the rankings of relevant items. The rankings of the relevant items change in a non-smooth way as a function of the predicted relevance scores and thus, RR_i is a non-smooth function over the model parameters. The non-smoothness of the RR measure makes it impossible to use standard optimization methods –such as gradient-based methods– to directly optimize RR_i. Inspired by recent developments in the area of learning to rank [6], we derive an approximation of $\mathbb{I}(R_{ik} < R_{ij})$ by using a logistic function:

$$\mathbb{I}(R_{ik} < R_{ij}) \approx g(f_{ik} - f_{ij}) \tag{2}$$

where $g(x) = 1/(1 + e^{-x})$, f_{ij} denotes the predictor function that maps the parameters from user i and item j to a predicted relevance score. The predictor function that we use in our model is the basic and widely-used factor model, expressed as:

$$f_{ij} = \langle U_i, V_j \rangle \tag{3}$$

where U_i denotes a d-dimensional latent factor vector for user i, and V_j a d-dimensional latent factor vector for item j. Even though a sophisticated approximation for the item rank was proposed in [6], it has not been deployed in practice. Notice that in the case of RR_i in Eq. (1), only $1/R_{ij}$ is actually in use. We thus propose to directly approximate $1/R_{ij}$ by another logistic function:

$$\frac{1}{R_{ij}} \approx g(f_{ij}) \tag{4}$$

which makes the basic assumption that the lower the item rank, the higher the predicted relevance score, i.e., $1/R_{ij}$ would approach to 1. Substituting Eq. (2) and (4) into Eq. (1), we obtain a smooth version of RR_i:

$$RR_i \approx \sum_{j=1}^{N} Y_{ij} g(f_{ij}) \prod_{k=1}^{N} \big(1 - Y_{ik} g(f_{ik} - f_{ij})\big) \tag{5}$$

Notice that although Eq. (5) is a smooth function with respect to the predicted relevance scores and thus the model parameters U and V, optimizing this function could still be practically intractable, due to its multiplicative nature. For example, the complexity of the gradient of Eq. (5) with respect to V_j (i.e., only for one item) is $O(N^2)$: the computational cost grows quadratically with the number of items N and for most recommender systems N is typically large. In the following, we present a lower bound of an equivalent variant of Eq. (5), for which we derive a computationally tractable optimization procedure.

3.2 Lower Bound of Smooth Reciprocal Rank

Suppose that the number of relevant items for user i in the given data collection is n_i^+. Given the monotonicity of the logarithm function, the model parameters that maximize Eq. (5) are equivalent to the parameters that maximize $\ln(\frac{1}{n_i^+} RR_i)$. Specifically, we have:

$$\begin{aligned} U_i, V &= \arg\max_{U_i,V} \{RR_i\} = \arg\max_{U_i,V} \{\ln(\frac{1}{n_i^+} RR_i)\} \\ &= \arg\max_{U_i,V} \{\ln\Big(\sum_{j=1}^{N} \frac{Y_{ij}}{n_i^+} g(f_{ij}) \prod_{k=1}^{N} \big(1 - Y_{ik} g(f_{ik} - f_{ij})\big)\Big)\} \end{aligned} \tag{6}$$

Based on Jensen's inequality and the concavity of the logarithm function, we derive the lower bound of $\ln(\frac{1}{n_i^+} RR_i)$ as below:

$$\begin{aligned} &\ln(\frac{1}{n_i^+} RR_i) \\ =&\ln\Big(\sum_{j=1}^{N} \frac{Y_{ij}}{\sum_{l=1}^{N} Y_{il}} g(f_{ij}) \prod_{k=1}^{N} \big(1 - Y_{ik} g(f_{ik} - f_{ij})\big)\Big) \\ \geq&\frac{1}{n_i^+} \sum_{j=1}^{N} Y_{ij} \ln\Big(g(f_{ij}) \prod_{k=1}^{N} \big(1 - Y_{ik} g(f_{ik} - f_{ij})\big)\Big) \\ =&\frac{1}{n_i^+} \sum_{j=1}^{N} Y_{ij} \Big(\ln g(f_{ij}) + \sum_{k=1}^{N} \ln\big(1 - Y_{ik} g(f_{ik} - f_{ij})\big)\Big) \end{aligned} \tag{7}$$

Note that in the derivation above we make use of the definition of n_i^+, i.e., $n_i^+ = \sum_{l=1}^{N} Y_{il}$. We can neglect the constant $1/n_i^+$ in the lower bound, and obtain a new objective function as:

$$L(U_i, V) = \sum_{j=1}^{N} Y_{ij} \big[\ln g(f_{ij}) + \sum_{k=1}^{N} \ln\big(1 - Y_{ik} g(f_{ik} - f_{ij})\big)\big] \tag{8}$$

We can take a close look at the two terms within the first summation. The maximization of the first term contributes to learning latent factors that promote relevant items. However, given one relevant item, e.g., item j, maximizing the second term turns to learning latent factors of all the other items (e.g., item k) in order to degrade their relevance scores. In sum, the two effects come together to promote and scatter the relevant items at the same time, the main characteristic

of the proposed *CLiMF*. In other words, *CLiMF* will lead to a recommendation where some but not all relevant items are at the very top of the recommendation list for a user. We notice that this behavior of *CLiMF* corresponds to the analysis of MRR optimization for a search result list [29], i.e., optimizing MRR results in diversifying ranked documents.

Taking into account the regularization terms that usually serve to control the complexity of the model (i.e. in order to avoid overfitting), and all the M users in the given data collection, we obtain the objective function of *CLiMF*:

$$F(U,V) = \sum_{i=1}^{M}\sum_{j=1}^{N} Y_{ij}\big[\ln g(U_i^T V_j) + \sum_{k=1}^{N}\ln\big(1 - Y_{ik} g(U_i^T V_k - U_i^T V_j)\big)\big] - \frac{\lambda}{2}(\|U\|^2 + \|V\|^2) \quad (9)$$

in which λ denotes the regularization coefficient, and $\|U\|$ denotes the Frobenius norm of U. Note that the lower bound $F(U,V)$ is much less complex than the original objective function in Eq. (5), and standard optimization methods, e.g., gradient ascend, can be used to learn the optimal model parameters U and V.

3.3 Optimization

We use stochastic gradient ascend to maximize the objective function in Eq. (9), i.e., for each user i, we optimize $F(U_i, V)$. The gradients of the objective for user i with respect to U_i and V_j can be computed as below:

$$\frac{\partial F}{\partial U_i} = \sum_{j=1}^{N} Y_{ij}\big[g(-f_{ij})V_j + \sum_{k=1}^{N}\frac{Y_{ik} g'(f_{ik} - f_{ij})}{1 - Y_{ik} g(f_{ik} - f_{ij})}\big(V_j - V_k\big)\big] - \lambda U_i \quad (10)$$

$$\frac{\partial F}{\partial V_j} = Y_{ij}\big[g(-f_{ij}) + \sum_{k=1}^{N} Y_{ik} g'(f_{ij} - f_{ik})\big(\frac{1}{1 - Y_{ik} g(f_{ik} - f_{ij})} - \frac{1}{1 - Y_{ij} g(f_{ij} - f_{ik})}\big)\big]U_i - \lambda V_j \quad (11)$$

where $g'(x)$ denotes the derivative of $g(x)$. Note that we have used a property of $g(x)$, namely, $g(-x) = g'(x)/g(x)$, in the derivation of Eq. (10) and (11) above to simplify the computation.

The learning algorithm for the *CLiMF* model is outlined in Algorithm 1. We analyze the complexity of the learning process for one iteration. By exploiting the data sparseness in Y, the computational complexity of the gradient in Eq. (10) is $O(d\tilde{n}^2 M + dM)$. Note that $\tilde{n}$ denotes the average number of relevant items across all the users. The complexity of computing the gradient in Eq. (11) is $O(d\tilde{n}^2 M + d\tilde{n}M)$. Hence, the complexity of the learning algorithm in one iteration is in the order of $O(d\tilde{n}^2 M)$. In the case that $\tilde{n}$ is a small number, i.e., $\tilde{n}^2 << M$, the complexity is linear to the number of users in the data collection. Note that we have $\tilde{n}M = S$, in which S denotes the number of non-zeros in the user-item matrix. The complexity of the learning algorithm is then $O(d\tilde{n}S)$. Since we usually have $\tilde{n} << S$, the complexity is $O(dS)$ even in the case that $\tilde{n}$ is large, i.e., being linear to the number of non-zeros (i.e. relevant observations in the data). In sum, our analysis shows that *CLiMF* is suitable for large scale use cases. Note that we also empirically verify the complexity of the learning algorithm in Section 4.4.

ALGORITHM 1: Learning Algorithm for *CLiMF*

Input: Training set Y, regularization parameter λ learning rate γ, and the maximal number of iterations $itermax$.
Output: The learned latent factors U, V.
for $i = 1, 2, \ldots, M$ **do**
  % Index relevant items for user i;
  $N_i = \{j | Y_{ij} > 0, 1 \le j \le N\}$;
end
Initialize $U^{(0)}$ and $V^{(0)}$ with random values, and $t = 0$;
repeat
  for $i = 1, 2, \ldots, M$ **do**
    % Update U_i;
    $U_i^{(t+1)} = U_i^{(t)} + \gamma \frac{\partial F}{\partial U_i^{(t)}}$ based on Eq. (10);
    for $j \in N_i$ **do**
      % Update V_j;
      $V_j^{(t+1)} = V_j^{(t)} + \gamma \frac{\partial F}{\partial V_j^{(t)}}$ based on Eq. (11);
    end
  end
  $t = t + 1$;
until $t \ge itermax$;
$U = U^{(t)}$, $V = V^{(t)}$

3.4 Discussion

We discuss the relationship between the proposed *CLiMF* and other state-of-the-art recommendation models, and present the insights that highlight the contribution of *CLiMF* to the area of CF when compared to other models.

Relation to CofiRank: CofiRank [30] was the first work that introduced learning to rank to address CF as a ranking problem. CofiRank makes use of structured estimation of a ranking loss based on NDCG, and learns the recommendation model by minimizing over a convex upper bound of the loss function. The major differences between *CLiMF* and CofiRank lie in two aspects: First, due to its foundation on the measure of NDCG, CofiRank suits to the scenarios where graded relevance data, e.g., ratings, are available from users to items, but might not be suitable for the scenarios with only binary relevance data, for which *CLiMF* is tailored. Second, CofiRank and *CLiMF* root in different classes of methods to achieve learning to rank [18, 31], such as the difference between SVM^{MAP} [33] and $SoftRank$ [27]. CofiRank exploits a convex upper bound of the structured loss function based on the evaluation metric NDCG, and then optimizes the upper bound. However, *CLiMF* first smooths the evaluation metric RR, and then optimizes the smoothed version of the metric via a lower bound.

Relation to CCF: Collaborative competitive filtering (CCF) [32] was proposed as an algorithm that not only exploits rated items from users, but also the candidate items (or *opportunities*) that were available for the users to choose. The key constraint introduced in CCF is that the utility (or relevance) of a rated item should be higher than any items that are in the candidate set but not rated/selected. *CLiMF* is similar to CCF in the sense that it also considers the rel-

ative pair-wise constraints in learning the latent factors, as shown in the second term with the summation in Eq. (8). However, *CLiMF* only requires relevant items, while CCF requires all the items in the candidate set, which are not usually available. In practice, CCF needs to include some unrated items together with the rated items to form the candidate set. In addition, CCF is not directly related to any evaluation metrics, while *CLiMF* is designed for MRR optimization.

Relation to OrdRec: OrdRec [14] is an ordinal model that formulates the probability that a rating predictor (a function of the model parameters, such as the latent factors) is equal to a known rating as the probability that the rating predictor falls in the interval of two parameterized scale thresholds corresponding to two adjacent rating values. OrdRec has a point-wise nature, i.e., it does not require any pair-wise computation between any rated/unrated items. Hence, it enjoys the advantage of a computational complexity that is linear to the data size, the same advantage attained by *CLiMF*. However, although OrdRec generally suits to scenarios with implicit feedback data, "count" information is necessary to extract the ordinals, i.e., the ordered preferences of users. For this reason, OrdRec may not be suitable for the scenarios with only binary relevance data. In addition, OrdRec has no direct relation to the ranking-oriented evaluation metrics.

Relation to BPR: BPR [21] models the pair-wise comparisons between positive and negative feedback data (in the scenarios with binary relevance data), and optimizes an objective that corresponds to Area Under Curve (AUC) optimization. BPR is similar to *CLiMF* in the sense that it also directly optimizes a smoothed version of an evaluation metric for binary relevance data, there are though two main differences. First, BPR requires a sampled set of negative feedback data, i.e., a set of unobserved items to be assumed as irrelevant to the users. However, *CLiMF* only requires the relevant items from the users. Second, while BPR aims at promoting all the relevant items, *CLiMF* particularly focuses on recommending a few but relevant items at top-k positions of the recommendation list, a goal which is attained by promoting and scattering relevant items at the same time, as shown in Eq. (8). Since BPR shares a close relationship with *CLiMF* in terms of modeling and application scenarios, we choose BPR as the main baseline to compare against in the experiments.

4. EXPERIMENTAL EVALUATION

In this section we present a series of experiments to evaluate *CLiMF*. We first describe the datasets used in the experiments and the setup. Then, we compare the recommendation performance of *CLiMF* with two baseline approaches in terms of providing only a few but relevant recommendations at the top positions of the recommendation list. Finally, we analyze the effectiveness and the scalability of the proposed *CLiMF* model.

We designed the experiments in order to address the following research questions:

1. Does the proposed *CLiMF* outperform alternative state-of-the-art algorithms, particularly when recommending just a few but relevant items at top-ranked positions?
2. Is the learning algorithm of *CLiMF* effective for increasing MRR to a local maximum?
3. Is *CLiMF* scalable for large-scale use cases?

Table 1: Statistics of the datasets

Dataset	Epinions	Tuenti
Num. non-zeros	346035	798158
Num. users	4718	11392
Num. friends/trustees	49288	50000
Sparseness	99.85%	99.86%
Avg. friends/trustees per user	73.34	70.06

4.1 Experimental Setup

4.1.1 Datasets

We conduct experiments using two social network datasets from Epinions[3] and Tuenti[4]. The Epinions dataset is publicly available[5], and contains trust relationship between 49288 users. The Epinions dataset represents a directed social network, i.e., if user i is a trustee of user j, user j is not necessary a trustee of user i. Most microblogging social networks are also directed, such as Twitter. For the purpose of our experiments, we exclude from the dataset the users who have less than 25 trustees. The second dataset collected from Tuenti, one of the largest social networks in Spain, represents an undirected social network, containing friendship between 50K users. Similar to the Epinions dataset, we also exclude the users with less than 25 friends. Note that in these two datasets, friends or trustees are regarded as "items" of users. The task is to generate friend or trustee recommendations to individual users. Statistics on the two datasets used in our experiments are summarized in Table 1.

4.1.2 Experimental Protocol and Evaluation Metrics

We separate each dataset into a training set and a test set under various conditions of user profiles. For example, the condition of "Given 5" denotes that for each user we randomly selected 5 out of her trustees/friends to form the training set, and use the remaining trustees/friends to form the test set. The task is to use the training set to generate recommendation lists for individual users, and the performance is measured according to the holdout data in the test set. We repeat the experiment 5 times for each of the different conditions of each dataset, and the performances reported are averaged across 5 runs. Again, we emphasize that in this work we only consider the observed items as being relevant to the user. Although this setting would underestimate the power of all the recommenders, the comparative results are still useful, since they can be regarded as the approximation of the lower limit of each recommender.

The main evaluation metric that we use in our experiments to measure the recommendation performance is MRR, the measure that is optimized in our model. In addition, we also measure the performance by precision at top-ranked items, such as precision at top-5 (P@5), which reflects the ratio of the number of relevant items in the top-5 recommended items. In order to emphasize the value of "less-is-more" recommendations, we also use the measure of 1-call at

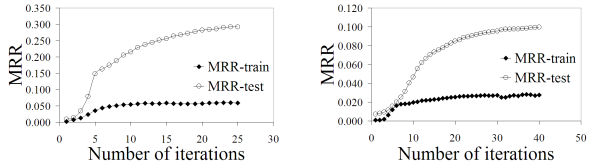
[3]http://www.epinions.com
[4]http://www.tuenti.com
[5]http://www.trustlet.org/wiki/Downloaded_Epinions_dataset

Table 2: Performance comparison of *CLiMF* and baselines on the Epinions dataset

	Given 5			Given 10			Given 15			Given 20		
	MRR	P@5	1-call@5	MRR	P@5	1-call@5	MRR	P@5	1-call@5	MRR	P@5	1-call@5
PopRec	0.142	0.035	0.166	0.127	0.032	0.134	0.117	0.032	0.136	0.131	0.048	0.210
iMF	0.154	0.059	0.225	0.143	0.059	0.236	0.155	0.063	0.231	0.153	0.059	0.226
BPR-MF	0.241	0.148	0.532	0.167	0.072	0.334	0.177	0.098	0.380	0.216	0.096	0.422
CLiMF	**0.292**	**0.216**	**0.676**	**0.233**	**0.092**	**0.392**	**0.248**	**0.127**	**0.496**	**0.239**	**0.110**	**0.448**

Table 3: Performance comparison of *CLiMF* and baselines on the Tuenti dataset

	Given 5			Given 10			Given 15			Given 20		
	MRR	P@5	1-call@5	MRR	P@5	1-call@5	MRR	P@5	1-call@5	MRR	P@5	1-call@5
PopRec	0.096	0.029	0.138	0.074	0.017	0.080	0.074	0.019	0.088	0.074	0.019	0.086
iMF	0.064	0.020	0.090	0.065	0.017	0.076	0.065	0.021	0.098	0.076	0.023	0.108
BPR-MF	0.096	0.030	0.142	0.075	0.025	0.116	0.075	0.020	0.090	0.076	0.021	0.106
CLiMF	**0.100**	**0.039**	**0.190**	**0.077**	**0.027**	**0.124**	**0.077**	**0.022**	**0.104**	**0.083**	**0.024**	**0.116**



(a) Epinions (b) Tuenti

Figure 1: Effectiveness of the learning algorithm for *CLiMF* under the "Given 5" condition for both datasets.

top-ranked items [7]. Specifically, 1-call at top-5 recommendations (1-call@5) reflects the ratio of test users who have at least one relevant item in their top-5 recommendation lists.

Finally, as revealed in recent studies from different recommender domains, popular items could highly dominate the recommendation performance [8, 25, 26]. We also notice this effect in our experiments, namely, recommending the most popular friends or trustees (i.e., those have the most friends or trusters) could already result in a high performance. For this reason, in our experiments we consider the top three most popular items as being irrelevant in order to reduce the influence from the most trivial recommendations [8, 25]. In other words, recommending any of the top three popular friends/ trustees has no contribution to any of the evaluation metrics.

4.1.3 Parameter Setting

We use one fold of randomly generated training-test sets of each dataset under the condition "Given 5" for the purpose of validation, which is used to tune parameters in CLiMF. The values of the parameters that yield the best performance on the validation set are: the regularization parameter $\lambda = 0.001$, the latent dimensionality $d = 10$ and the learning rate $\gamma = 0.0001$.

4.2 Performance Comparison

We compare the performance of *CLiMF* with three baselines, PopRec, iMF and BPR, which are described below:

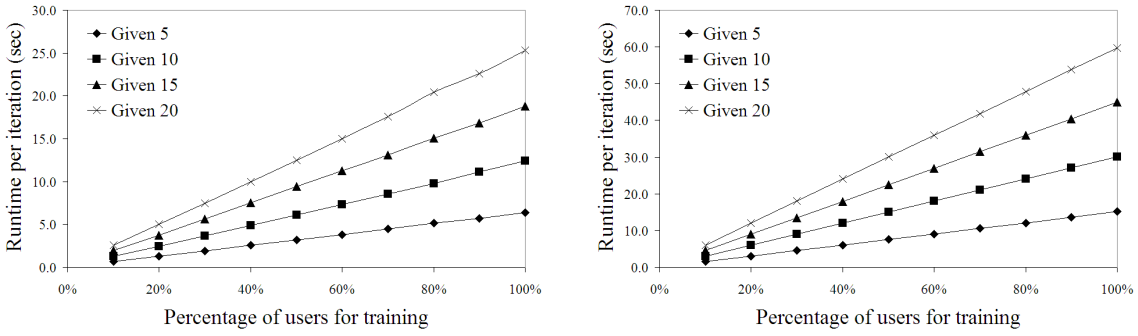
- **PopRec**. A naive baseline that recommends a user to be a friend or trustee in terms of her popularity, i.e., the number of friends or trusters she has in the given training set. The more friends or trusters the user has, the higher her position in the recommendation list. Note that it is a non-personalized recommendation approach: for any target user, the recommendations are always the same.

- **iMF**: A state-of-the-art matrix factorization technique for implicit feedback data by Hu et al. [12], as discussed in Section 2.

- **BPR-MF**. Bayesian personalized ranking (BPR) represents the state-of-the-art optimization framework of CF for binary relevance data [21]. BPR-MF represents the choice of using matrix factorization (MF) as the learning model with BPR optimization criterion. Note that the implementation of this baseline is done with the publicly available software MyMediaLite [10]. The relevant parameters, such as the regularization coefficients and the number of iterations, are tuned on the validation sets, which are the same sets that were used for tuning the *CLiMF* model.

The recommendation performances of *CLiMF* and the baseline approaches on the Epinions and the Tuenti datasets are shown in Table 2 and Table 3, respectively.

Three main observations can be drawn from the results: First, the proposed *CLiMF* model *significantly* outperforms the three baselines in terms of MRR across all the conditions and the two datasets. Note that in our experiments, the statistical significance is measured based on the results from individual test users, according to a Wilcoxon signed rank significance test with $p<0.01$. This result corroborates that *CLiMF* achieves the goal that was designed for and optimizes the value of the reciprocal rank for the recommendations to the individual users. Second, *CLiMF* also achieves a *significant* improvement over the baselines in terms of P@5 and 1-call@5 across all the conditions and the two datasets. The improvement of P@5 indicates that by optimizing MRR, *CLiMF* also improve the quality of recommendations among the top-ranked items. In addition, the improvement of 1-call@5 supports that *CLiMF* particularly contributes to providing valuable recommendations at the top-k positions, i.e., raising the chance that users would receive at least one relevant recommendation among just a few top-ranked items. Compared to BPR, where AUC is optimized, *CLiMF* succeeds in enhancing the top-ranked performance by optimizing MRR, the top-biased metric. As can be also seen from the results, iMF performs worse than



(a) Epinions (b) Tuenti

Figure 2: Scalability analysis of *CLiMF* in terms of the number of users in the training set

both BPR and *CLiMF* in all the conditions of the Epinions dataset and in most of the conditions of the Tuenti dataset. The reason might be that iMF is particularly designed for implicit feedback datasets with the "count" information as mentioned in Section 2, while it may not be suitable for the scenarios with only binary relevance data. Third, relatively large improvements are attained in terms of most of the metrics when users have a low number of known friends/trustees, i.e., the case of "Given 5". This result suggests that *CLiMF's* key mechanism of scattering relevant items could be particularly beneficial for recommendation scenarios under very high data sparseness.

Hence, we give a positive answer to our first research question.

4.3 Effectiveness

The second experiment investigates the effectiveness of the proposed learning algorithm for *CLiMF*, as presented in Section 3.3. Figures 1 (a) and (b) show the evolution of MRR with each iteration –as measured in both the training and the test sets– under the "Given 5" condition for the Epinions and Tuenti datasets, respectively. We can see that both MRR measures gradually increase with each iteration and convergence is reached after a few iterations, i.e., nearly after 20 iterations on the Epinions dataset and 30 iterations on the Tuenti dataset. This observation indicates that *CLiMF* effectively learns from the training set latent factors of users and items that optimize reciprocal rank, which consequently also contributes to improving MRR in the test set.

With this experimental result, we give a positive answer to our second research question.

4.4 Scalability

The last experiment investigates the scalability of *CLiMF*, by measuring the training time that is required for the training set at different scales. First, as analyzed in Section 3.3, the computational complexity of *CLiMF* is linear to the number of users in the training set when the average number of friends/trustees per user is fixed. To demonstrate the scalability, we use different numbers of users in the training set under each condition: we randomly select from 10% to 100% users in the training set and their known friends/trustees as the training data for learning the latent factors. The results on the Epinions dataset and the Tuenti dataset are shown in Fig. 2(a) and 2(b), respectively. We can observe that for both datasets, the computational time under each condition increases almost linearly to the increase of the number of users. Second, as also discussed in Section 3.3, the computational complexity of *CLiMF* could be further approximated to be linear to the amount of known data (i.e., non-zero entries in the training user-item matrix). To demonstrate this, we examine the runtime of the learning algorithm against different scales of the training sets under different "Given" conditions. For example, under the "Given 5" condition of the Epinions dataset, there are $5 \times 4718 = 23590$ non-zeros in the training set. The result is shown in Fig. 3, from which we can observe that the average runtime of the learning algorithm per iteration increases almost linearly as the number of non-zeros in the training set increases.

The observations from this experiment allow us to answer our last research question positively.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a new CF approach, *CLiMF*, that learns latent factors of users and items by directly maximizing MRR. The *CLiMF* is designed to improve the performance of top-k recommendations for usage scenarios with only binary relevance data. We have demonstrated in our experiments that *CLiMF* offers significant improvements over a naive and two state-of-the-art baselines in two social network datasets. We have also experimentally validated that *CLiMF's* learning algorithm is effective for MRR optimization, and has linear computational complexity to the size of the known data, and thus is scalable for large scale use cases.

Future work involves a few interesting directions. First, we would like to extend our *CLiMF* model to suit domains with explicit feedback data, e.g., ratings. Second, it is also interesting to experimentally investigate the impact of *CLiMF* on the recommendation diversity, by exploiting external information resources, such as the categories of items. Third, we are also interested in investigating recommendation models that optimize other evaluation measures, and in exploring the impact of optimizing different measures on various aspects of recommendation performance [29].

6. ACKNOWLEDGEMENTS

This work is funded as part of a Marie Curie Intra European Fellowship for Career Development (IEF) award (CARS, PIEF-GA-2010-273739) held by Alexandros Karatzoglou.

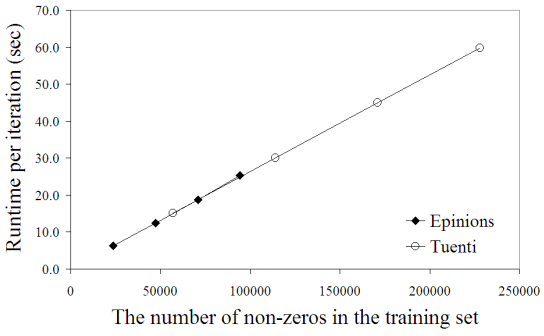


Figure 3: Scalability analysis of *CLiMF* in terms of the scale of the training data

7. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[2] D. Agarwal and B.-C. Chen. Regression-based latent factor models. KDD '09, pages 19–28. ACM, 2009.

[3] S. Balakrishnan and S. Chopra. Collaborative ranking. WSDM '12, pages 143–152. ACM, 2012.

[4] C. J. C. Burges, R. Ragno, and Q. V. Le. Learning to Rank with Nonsmooth Cost Functions. In *NIPS*, pages 193–200. MIT Press, 2006.

[5] S. Chakrabarti, R. Khanna, U. Sawant, and C. Bhattacharyya. Structured learning for non-smooth ranking losses. KDD '08, pages 88–96. ACM, 2008.

[6] O. Chapelle and M. Wu. Gradient descent optimization of smoothed information retrieval metrics. *Inf. Retr.*, 13:216–235, June 2010.

[7] H. Chen and D. R. Karger. Less is more: probabilistic models for retrieving fewer relevant documents. SIGIR '06, pages 429–436. ACM, 2006.

[8] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. RecSys '10, pages 39–46. ACM, 2010.

[9] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22:143–177, January 2004.

[10] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Mymedialite: a free recommender system library. In *Proceedings of the fifth ACM conference on Recommender systems*, RecSys '11, pages 305–308, New York, NY, USA, 2011. ACM.

[11] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22:89–115, January 2004.

[12] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. ICDM '08, pages 263–272, 2008.

[13] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, August 2009.

[14] Y. Koren and J. Sill. Ordrec: an ordinal model for predicting personalized item rating distributions. RecSys '11, pages 117–124. ACM, 2011.

[15] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7:76–80, 2003.

[16] N. N. Liu and Q. Yang. Eigenrank: a ranking-oriented approach to collaborative filtering. SIGIR '08, pages 83–90. ACM, 2008.

[17] N. N. Liu, M. Zhao, and Q. Yang. Probabilistic latent preference analysis for collaborative filtering. CIKM '09, pages 759–766. ACM, 2009.

[18] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[19] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. ICDM '08, pages 502–511, 2008.

[20] L. Pizzato, T. Rej, T. Chung, I. Koprinska, and J. Kay. Recon: a reciprocal recommender for online dating. RecSys '10, pages 207–214. ACM, 2010.

[21] S. Rendle, C. Freudenthaler, Z. Gantner, and S.-T. Lars. Bpr: Bayesian personalized ranking from implicit feedback. UAI '09, pages 452–461. AUAI Press, 2009.

[22] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. CSCW '94, pages 175–186. ACM, 1994.

[23] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. volume 20 of *NIPS '08*, 2008.

[24] Y. Shi, M. Larson, and A. Hanjalic. List-wise learning to rank with matrix factorization for collaborative filtering. RecSys '10, pages 269–272. ACM, 2010.

[25] Y. Shi, P. Serdyukov, A. Hanjalic, and M. Larson. Personalized landmark recommendation based on geotags from photo sharing sites. ICWSM '11, pages 622–625. AAAI, 2011.

[26] H. Steck. Item popularity and recommendation accuracy. RecSys '11, pages 125–132. ACM, 2011.

[27] M. Taylor, J. Guiver, S. Robertson, and T. Minka. Softrank: optimizing non-smooth rank metrics. WSDM '08, pages 77–86. ACM, 2008.

[28] E. M. Voorhees. The trec-8 question answering track report. In *TREC-8*, 1999.

[29] J. Wang and J. Zhu. On statistical analysis and optimization of information retrieval effectiveness metrics. SIGIR '10, pages 226–233. ACM, 2010.

[30] M. Weimer, A. Karatzoglou, Q. Le, and A. Smola. Cofirank - maximum margin matrix factorization for collaborative ranking. NIPS '07, pages 1593–1600, 2007.

[31] J. Xu, T.-Y. Liu, M. Lu, H. Li, and W.-Y. Ma. Directly optimizing evaluation measures in learning to rank. SIGIR '08, pages 107–114. ACM, 2008.

[32] S.-H. Yang, B. Long, A. J. Smola, H. Zha, and Z. Zheng. Collaborative competitive filtering: learning recommender using context of user choice. SIGIR '11, pages 295–304. ACM, 2011.

[33] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. SIGIR '07, pages 271–278. ACM, 2007.