Metric Factorization: Recommendation beyond Matrix Factorization

Preprint · June 2018

6 authors, including:

Shuai Zhang
UNSW Sydney
18 PUBLICATIONS **55** CITATIONS
SEE PROFILE

Lina Yao
UNSW Sydney
141 PUBLICATIONS **698** CITATIONS
SEE PROFILE

Yi Tay
Nanyang Technological University
31 PUBLICATIONS **126** CITATIONS
SEE PROFILE

Xiwei Xu
The Commonwealth Scientific and Industrial Research Organisation
61 PUBLICATIONS **457** CITATIONS
SEE PROFILE

Metric Factorization: Recommendation beyond Matrix Factorization

Shuai Zhang
UNSW and Data61, CSIRO
Sydney, NSW, Australia
shuai.zhang@student.unsw.edu.au

Lina Yao
University of New South Wales
Sydney, NSW, Australia
lina.yao@unsw.edu.au

Yi Tay
Nanyang Technological University
Singapore
ytay017@e.ntu.edu.sg

Xiwei Xu
Data61, CSRIO
Sydney, NSW, Australia
Xiwei.Xu@data61.csrio.au

Xiang Zhang
University of New South Wales
Sydney, NSW, Australia
xiang.zhang3@student.unsw.edu.au

Liming Zhu
Data61, CSRIO
Sydney, NSW, Australia
Liming.Zhu@data61.csrio.au

ABSTRACT

In the past decade, matrix factorization has been extensively researched and has become one of the most popular techniques for personalized recommendations. Nevertheless, the dot product adopted in matrix factorization based recommender models does not satisfy the inequality property, which may limit their expressiveness and lead to sub-optimal solutions. To overcome this problem, we propose a novel recommender technique. We assume that users and items can be placed in a low dimensional space and their explicit closeness can be measured using Euclidean distance which satisfies the inequality property. To demonstrate its effectiveness, we further designed two variants with one for rating estimation and the other for personalized item ranking. Extensive experiments on a number of real-world datasets show that our approach outperforms existing state-of-the-art by a large margin on both rating prediction and item ranking tasks.

CCS CONCEPTS

• **Information systems → Recommender systems**;

KEYWORDS

Recommender Systems; Metric Learning; Matrix Factorization.

1 INTRODUCTION

Learning low-rank latent structure from preference matrices forms the bedrock of many decades of recommender system research [14, 19, 28, 32]. This concept, widely known as Matrix Factorization [28], has established itself as one of the more popular techniques for recommendation. Hinging upon the simplicity and effectiveness of the inner product, this family of techniques have demonstrated reasonable success, with widespread adoption spurred on by successful entries in notable high-profile competitions such as the Netflix Prize[1].

On the other hand, a competing branch, relying on computation and learning in metric vector space [15, 37, 40], has not only demonstrated rivaling simplicity but also effectiveness. A core argument against factorization models is that inner product violates the triangle inequality, which is highly essential for capturing fine-grained similarity between user and items. To this end, metric learning approaches, largely based on Euclidean distances as a similarity metric, is consistently argued to be more adept and expressive.

This paper is primarily concerned with bridging both paradigms. For the first time, we explore the notion of *'unraveling metric vector spaces'*, i.e., learning low-rank structure in metric space via factorization. To this end, our proposed approach, Factorized Metric Learning (FML) can be interpreted as factorizing preference matrix comprising distances instead of rating scores. This is easily different from standard matrix factorization, which typically encodes user-item similarity via rating score.

The primary motivator behind our approach is to combine the best of both worlds, which consequently, mitigate the inherent weaknesses of each paradigm. More concretely, pertaining to MF, its performance may be hindered by the simple choice of interaction method - the inner product product [13, 33]. It is known that dot product does not satisfy the triangle inequality [25][2], limiting the expressiveness of matrix factorization, therefore resulting in sub-optimal solutions. Detailed explanations can be found in [13, 15]. Moreover, empirical studies also show that matrix factorization is prone to over-fitting for large size of latent factors, which substantially restrict the model flexibility and capability [13, 42].

Metric learning, popularized by a recent work *Collaborative Metric Learning* (CML) has demonstrated promising capability in tackling this issue. However, it possesses it's own set of problems, i.e., over-congestion in vector space [37]. While it learns similarity clusters, we hypothesize that the learned clusters may also suffer from over-congestion, resulting in sub-optimality. We further demonstrate this phenomena in our qualitative experiments.

To mitigate these above mentioned problems, we propose a novel technique: **Factorized Metric Learning** (FML). The key intuition is to firstly convert the preference into distance and then replace the dot product with Euclidean distance. FML considers the recommendation problem from a position and distance perspective and directly factorizes the distance matrix into users and items

[1] https://www.netflixprize.com/

[2] This is defined as "the distance between two points cannot be larger than the sum of their distances from a third point" [38].

conference, 2018,
© 2018 Copyright held by the owner/author(s).
ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

dense embeddings. It can not only avoid the aforementioned shortcomings, but also is applicable to both rating prediction and item ranking tasks. Recommendations are made based on estimated distances. As such, this can be interpreted as *unraveling* distances with factorization. All in all, the main contributions of this work are summarized as follows:

- We proposed a novel technique, Factorized Metric Learning (FML) in which we explore the notion of finding low-rank structure in metric vector spaces. In our approach, users and items are represented as points in a multi-dimensional coordinate system (i.e., metric vector space). Recommendations are made based on user and item closeness, defined in metric space. However, unlike other metric learning approaches, the key novelty behind our model is that it factorizes the metric space.
- We specify two variants of FML to solve the two classic and well-established recommendation tasks: rating prediction and item ranking. Our model can effectively learn the positions of users and items in both settings.
- Extensive experiments on a wide spectrum of large scale datasets demonstrate that our model outperforms the state-of-the-art models by a large margin in terms of both rating estimation and item ranking tasks. This shows that our conceptually simple model can be highly effective.

2 BACKGROUND

Matrix Factorization for Recommendation. Matrix factorization is one of the most effective methods for item recommendation. The first version of matrix factorization for recommender task is designed by Simon Funk[3] in Netflix contest for rating prediction. Later studies improved MF and proposed many variants. For example, Koren et al. [19] introduced user and item biases to model user and item specific features. Salakhutdinov et al. [28, 29] interpret MF into a probabilistic graphical model to alleviate the sparsity and in-balance in real-world datasets. Matrix factorization can also be generalized to solve personalized item ranking problem. Two classic top-N recommendation models based on matrix factorization are: Bayesian Personalized Ranking (BPR) [26] and Weighted Regularized matrix factorization (WRMF) [16]. BPR tackles the item ranking problem from a Bayesian perspective, and it aims to push unobserved user item pairs away from observed user item pairs. WRMF is another effective approach for item ranking. WRMF uses implicit binary feedback as training instances and considers all entries (including unobserved entries) of the user item interaction matrix. WRMF has a confidence value used to control the weight of negative and positive entries.

Neural Models for Recommendation. As aforementioned, despite the success of MF, its capabilities are constrained by dot product. To this end, several attempts are made to overcome this problem. One approach is to introduce non-linearity to matrix factorization [44, 45]. Dziugaite et al. [5] proposed a neural network generalization of matrix factorization by modeling the user-item relationships with non-linear neural networks. The basic idea is to apply non-linear activation functions over the element-wise product of user and item latent factors. He et al. [13] followed this idea and proposed the neural collaborative filtering (NeuMF) model for personalized ranking task. NeuMF consists of a generalized MF and a multilayer perceptron. It treats the one-class collaborative filtering problem as a classification task and optimizes the network with a cross-entropy loss. Wu et al. [42] propose using denoising autoencoder to introduce non-linearity for interaction modelling. Incorporating side information with neural networks is also viable [10–12, 24, 35, 36, 39, 46, 47]. Nonetheless, these studies mainly focus on overcoming the limitation of dot product and we leave the side information modelling as a future work. A more comprehensive review on this topic can be found in [44].

Metric Learning for Recommendation. Another promising attempt is to directly adopt a metric which satisfies the axiom of triangle inequality. Collaborative metric learning (CML) [15] is such a method which generalizes metric learning to collaborative filtering. CML follows the idea of the largest margin nearest neighbour algorithm (LMNN) [40]. LMNN aims to estimate a linear transformation to formulate a distance metric that minimizes the expected kNN classification errors. LMNN consists of two critical operations: *pull* and *push*. The pull operation acts to pull instances in the same class closer, while the push operation acts to push different labeled instances apart. Strictly, CML does not learn the transformation function but the user vectors and item vectors. It only has a push term, which means that CML can only push items that the user dislikes away while does not provide a strategy to pull items the user likes closer. Hsieh et al. [15] mention that the loss function of CML would pull positive items when encountering imposters. Nevertheless, this kind of indirect pull force is weak compared with that in LMNN, which might cause over-congestion (see Figure 4-Right) and push possible recommendation candidates too far away, thus, leading to sub-optimal solutions.

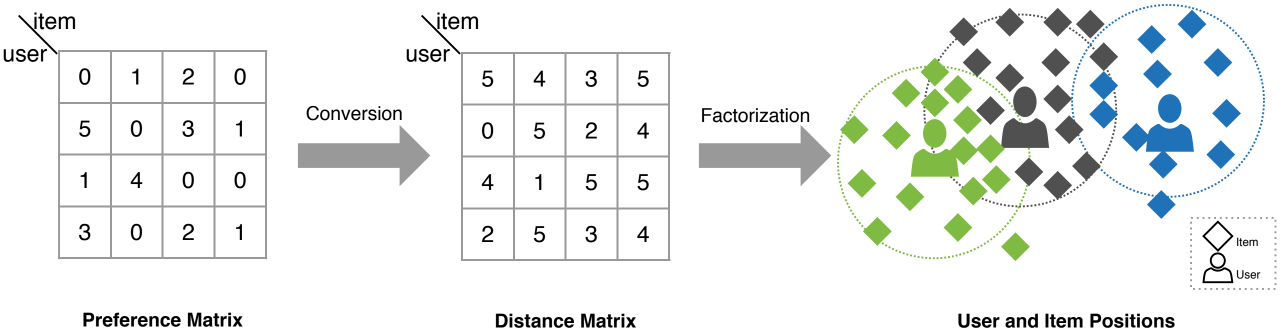
3 PROPOSED METHODOLOGY

In this section, we introduce our *Factorized Metric Learning* approach. Figure 1 illustrates a brief high-level overview of the FML. Suppose we have M users and N items, the preference (rating or implicit interaction) matrix is represented by $R \in \mathcal{R}^{M \times N}$. R_{ui} indicates the preference of user u to item i. The value of the R can be explicit ratings such as rating score or implicit feedback. Explicit ratings can be utilized for rating estimation for unknown entries, while implicit feedback is usually used for personalized ranking. These two tasks are commonplace in recommendation area, so we do not overstate them [27].

3.1 Preference as Distance

We aim to unravel the metric vector space to learn user and item positions by factorization. Matrix factorization learns user and item latent factors via factorizing the preference matrix (from explicit/implicit feedback) into dot product. Preference matrix can also be viewed as similarity matrix. Since similarity and distance are two opposite concepts, we need to convert the user preference to distance at first.

[3] http://sifter.org/simon/journal/20061211.html



0	1	2	0
5	0	3	1
1	4	0	0
3	0	2	1

5	4	3	5
0	5	2	4
4	1	5	5
2	5	3	4

Figure 1: A simplified illustration of FML. It mainly has two steps: First, convert preference matrix (rating or interaction) to distance matrix via equation 1. Second, factorize the distance matrix into user and item positions in a metric vector space.

Here, we give a simple yet effective principle. The distance matrix is achieved via the following equation:

$$\textbf{Distance}(u,i) = \textbf{Max Similarity} - \textbf{Similarity}(u,i) \tag{1}$$

Maximum similarity is the maximum value of ratings (e.g., 5) or implicit feedback (e.g., 1). Through this flip operation, we manage to convert preference (similarity) to distance while keeping its distribution. This conversion can be applied to both explicit and implicit feedback. Distance is a numerical measurement of how far apart objects are. Distance function should satisfy four critical conditions: non-negativity, identity of indiscernibles, symmetry and triangle inequality [40, 43]. There are many distance functions that are used in different application domains such as discrete distance, Euclidean distance, Graph distance. In the Euclidean space $\mathcal{R}^k$, distance between two points are usually measured by Euclidean distance (or ℓ_2-norm distance). It has been widely used in machine learning, sensor networks, geometry, face recognition, etc., due to its simple form and useful properties [3, 8, 22]. In practice, squared Euclidean distance is usually adopted to avoid the trouble of computing the square root.

Suppose the positions of user and item in a metric vector space are denoted by $P_u \in \mathcal{R}^k$ and $Q_i \in \mathcal{R}^k$. We measure the distance between user and item with squared Euclidean Distance:

$$\mathcal{D}(u,i) = \| P_u - Q_i \|_2^2 \tag{2}$$

Figure 1 simply illustrates the procedure of FML. First, we get the distance matrix from the preference matrix with Equation 1, then we factorize the distance matrix and learn the positions of users and items. After that, if required, we can easily recover every entry of the preference matrix and make recommendations.

3.2 Factorizing Distance for Item Ranking

Personalized ranking is an important major task in recommender systems when only implicit feedback is available. In many real world applications, implicit data such as purchase records, listen to tracks and clicks are more easily and readily available than explicit feedback, which leads to modelling implicit feedback a primary focus. We follow former studies and define implicit feedback as binary values, with 1 indicating like and 0 for other cases [13, 18].

Firstly, we convert the implicit feedback to distance using the following transformation:

$$Y_{ui} = a \cdot (1 - R_{ui}) \tag{3}$$

Since R_{ui} equals to either 0 or 1, the distance $Y_{ui} = a$ if $R_{ui} = 0$ and $Y_{ui} = 0$ if $R_{ui} = 1$, making it very flexible to control the user and item distances.

For ranking task, it is usually beneficial to take the unobserved interactions (negative samples) into consideration [13, 16]. For example, Bayesian personalized ranking and collaborative metric learning are trained in pairwise manner by sampling a negative item for each observed interaction. WRMF and NeuMF adopt a pointwise learning approach but also consider negative items. In this work, a pointwise loss is adopted as we want to directly factorize the distance into user and item embeddings.

$$\mathcal{L}(P,Q) = \sum_{(u,i)\in R} c_{ui}(Y_{ui} - \mathcal{D}(u,i))^2 \tag{4}$$

Here, all unobserved interactions are considered. c_{ui} is a confidence value and it is defined as: $c_{ui} = 1 + \alpha w_{ui}$, where w_{ui} represents the observation of implicit feedback. We treat the frequency of taking implicit actions as the observation, for example, if a user browses the item twice, then $w_{ui} = 2$. For large numbers, we can also convert it to log-scale like [16]. Since this information is usually absent in publicly available dataset, we set w to R_{ui} (0 or 1). Regularization strategy will be introduced in Section 3.4. Our model can not only force users and their preferred items closer but also push unpreferred items away even though we adopt a pointwise training scheme. Unlike most metric learning based models, which enforce imposters outside of the user territory of preference by some finite margin, the confidence mechanism in this approach provides possibility for negative items to invade user territory, which is beneficial for recommendation task as it can work as a filter to select items from negative candidates for recommendation. Another important characteristic of our model is that it can indirectly cluster users who share a large amount of items together. This property makes it possible to capture the relationships between neighbourhoods which are also significant for item recommendation [1].

3.3 Factorizing Distance for Rating Prediction

As aforementioned, our method can be also applied to rating estimation. For rating prediction, it is enough and more efficient to just consider observed interaction data. Let $\mathcal{K}$ denote the set of observed rating data. First, we convert the rating matrix R into distance with the following equation:

$$Y_{ui} = R^{max} - R_{ui} \tag{5}$$

Where R^{max} is the maximum rating score. If $R^{max} = 5$ and the rating score of 3, then the distance $Y_{ui} = 5 - 3 = 2$.

Same as matrix factorization [19], individual effects of users or items also matter. For example, some items tend to receive higher ratings; some users tend to give low rating scores. As such, we integrate the global user and item bias terms into our approach for rating estimation. The final scoring function is formulated as follows:

$$\hat{Y} = \mathcal{D}(u, i) + b_u + b_i + \mu \tag{6}$$

Here, $\hat{Y}$ denotes the predicted distances. b_u and b_i are the user and bias terms, respectively; μ is the global bias which equals to the mean distance constructed from training data. In general, we can add a hyper-parameter τ to scale μ to a more appropriate value.

Another important aspect we would like to consider is the reliability and stability of the rating data. Most rating prediction algorithms ignore the noise of ratings and assume that all ratings can be treated as ground truth. Nonetheless, not all observed ratings deserve the same weight [19]. For example, some users might give two different scores when being asked to rate the same item twice at different time [17]. Former studies [2, 17] suggest that extreme ratings (e.g., 1 and 5) are more reliable than moderate ratings (e.g, 2, 3 and 4). To alleviate this situation, we propose adding a confidence value c_{ui} for each rating score and get the following loss function:

$$\mathcal{L} = \sum_{(u,i)\in\mathcal{K}} c_{ui}(Y_{ui} - \hat{Y}) \tag{7}$$

Note that the confidence c_{ui} can represent many factors. Inspired by the conclusion of [17], we design a novel confidence mechanism to assign higher confidence values to more reliable ratings.

$$c_{ui} = 1 + \alpha \cdot g(R_{ui} - R^{max}/2) \tag{8}$$

where $g(\cdot)$ can be absolute, square, or even logarithm functions. α (confidence level) is a hyper-parameter to control the magnitude of the confidence. This confidence mechanism ensures extreme ratings to get higher confidence. It is possible to replace this confidence strategy with other alternatives.

3.4 Regularization

Traditional matrix factorization usually uses ℓ_2-norm regularization on latent factors and biases to avoid over-complex solutions. Here, we introduce another two regularization options: norm clipping and dropout.

Norm Clipping. For P and Q, ℓ_2 norm regularization is undesirable as it will push users and items close to the origin. Instead of minimizing the ℓ_2 regularization, we relax the constraints to the Euclidean ball and perform ℓ_2 norm clipping after each update.

$$\| P_u \|_2 \leq l, \| Q_i \|_2 \leq l \tag{9}$$

where l control the size of the Euclidean ball. These two constraints work as regularization to restrict the value of U_u and V_i in ℓ_2-norm unit ball so that the data points will not spread too widely [6, 15], which is beneficial to overcome the curse of dimensionality problem. This operation is generally performed after updating the parameters in every iteration.

Dropout. Dropout is a simple but useful approach for tackling overfitting problem in neural networks [34]. The main idea is to drop some neurons during training stage to avoid co-adaptation between neural units. Here, we propose a novel dropout strategy for Euclidean distance. The final Euclidean distance is the addition of the distance of each dimension. To prevent the co-adaptations among dimensions, we propose randomly dropping some dimensions and computing the overall distance with the following equation:

$$\| P_u - Q_i \|_2^2 = |P_{u1} - Q_{i1}|^2 + \underbrace{|P_{u2} - Q_{i2}|^2}_{drop} + \underbrace{|P_{u3} - Q_{i3}|^2}_{drop} + ... + |P_{uj} - Q_{ij}|^2 + ... + \underbrace{|P_{uk} - Q_{ik}|^2}_{drop}$$

in the above example, we remove the second, third till k^{th} dimension, and thus these dimensions will not contribute to the distance prediction in the specific epoch. The dropout dimensions are subject to change *in each epoch*. Note that this dropout operation is only carried out during the training period.

In the rating prediction model, we still utilize ℓ_2 norm with regularization rate λ to regularize the user and item biases.

3.5 Optimization and Recommendation

We optimize both rating estimation and ranking learning models with Adagrad [4]. Adagrad adapts the step size to the parameters based on the frequencies of updates. As such, it reduces the demand to tune the learning rate. In the recommendation stage, we firstly compute the distance $\hat{Y}_{ui}$ between users and items. For rating prediction, we reconstruct the ratings using the following rule: $\hat{R}_{ui} = R^{max} - \hat{Y}_{ui}$. For example, if $\hat{Y}_{ui} = 4$ and $R^{max} = 5$, the predicted rating score will be 1. For item ranking, we directly rank the distance in descending order and items that close to the target user will be recommended.

3.6 Model Complexity

The complexity of FML on rating prediction is linearly proportional to the number of ratings. It can be trained in $O(|\mathcal{K}|k)$, where k is the dimension of latent factors and $|\mathcal{K}|$ is the number of observed ratings. For item ranking, since all entries of the interaction matrix are involved, the model complexity is $O(|R|k)$. Concretely, it takes about $50s$ to get a satisfactory recommendation performance for rating estimation on Movielens 100K (on a NVIDIA TITAN XPascal GPU); For item ranking on FilmTrust, the computation time for each epoch is about $15s$, and it usually takes less than 30 epochs to achieve a satisfactory result.

4 EXPERIMENTS

Since rating prediction and item ranking are usually investigated separately with different evaluation metrics, we evaluate the proposed approach on these two tasks individually. We empirically evaluate the proposed methodology mainly to answer the following three research questions:

- **RQ1** Can FML outperform neural network and metric learning approaches in terms of item ranking?
- **RQ2** Does FML achieve more accurate rating estimation than MF and neural network based models?
- **RQ3** How do the hyper-parameters impact the model performances?

4.1 Evaluation for Item Ranking

Datasets Description. To evaluate the ranking performance of FML on implicit feedback, we conduct experiments on the following four real-world datasets. Implicit feedback is constructed in the same way as [13].

- **Yahoo Research Dataset**. It consists of two datasets: Yahoo movie and Yahoo music. Both of them are provided by Yahoo Research Alliance Webscope program[4]. The music preference dataset is collected from Yahoo Music services.
- **FilmTrust**. FilmTrust is crawled from a film sharing and rating website by Guo et al. [9]. The trust information is not used in this paper.
- **EachMovie**. EachMovie[5] is a benchmark movie dataset that is widely used for recommendation evaluation.

Evaluation Metrics. To evaluate the ranking accuracy and quality, we adopt five widely used metrics: Recall@n, Precision@n, Mean Average Precision (MAP), Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain (DNCG). In practice, making items that interest target users more rank higher will enhance the quality of recommendation lists. MAP, MRR and NDCG are three rank-aware measurements with which higher ranked positive items are prioritized. Thus they are suitable for assessing the quality of ranking list [30]. MRR cares about the single highest-ranked relevant item. NDCG is determined by Discounted Cumulative Gain and Ideal Discounted Cumulative Gain. All the evaluation metrics follow the implementation of a well-known open source recommendation project[6], same implementation can also be found in LibRec[7]. Detail definitions are omitted for the sake of conciseness, we refer the reader to [31] for more details.

Baselines. We compare FML with the following classic and recent state-of-the-art baselines:

- **BPR** [26]. It is a generic pairwise optimization criterion and works well with MF. The BPR optimization criterion aims to maximize the differences between negative and positive samples.
- **WRMF** [16]. It is an effective ranking model for implicit feedback. It uses dot product to model the user item interaction patterns.
- **NeuMF** [13]. NeuMF combines multi-layered preceptron with generalized matrix factorization and computes the ranking scores with a neural layer instead of dot product. It treats the ranking problem as a classification task and optimizes the model by minimizing the cross entropy loss.
- **CDAE** [42]. CDAE learns user and item distributed representations from interaction matrix with autoencoder. Denoising technique is utilized to improve generalization and prevent overfitting. Here, we use logistic loss to optimize the network as it is reported to achieve the best performances in the original paper.
- **CML** [15]. CML is a competitive baseline. Here, we use two types of loss: pairwise hinge loss and weighted approximate-rank pairwise loss (WAPR) to train this model. Note that the WARP loss is computational intensive due to the pre-computations for negative samples [41].

In total, BPR and WRMF are matrix factorization based models; NeuMF and CDAE are neural network based models; CML-PAIR and CML-WARP are metric learning based models.

Implementation Details. We adopt the implementation of Mymedialite[8] for BPR and WRMF, and the authors' released source code for NeuMF [9]. We implemented all other models with Tensorflow. We test our model with five random splits by using 80% of the interaction data as training set and the remaining 20% as test set, and report the average results. Hyper-parameters are determined with grid search to ensure all baselines to achieve their best performance. The dimensions of user and item vectors in FML and CML are tuned amongst $\{80, 100, 120, 150, 200\}$. Learning rate of FML is set to 0.05. The margin of CML-pair and CML-WARP is amongst $\{0.5, 1.0, 1.5, 2.0\}$. The distance scale factor a (Eq. 3) is set to: 2.25 for FilmTrust, 2.0 for YahooMovie and EachMovie, 0.5 for YahooMusic. w_{ui} is set to R_{ui} for all datasets. The confidence level α is set to 4 for FilmTrust and 1 for other three datasets. Dropout does not influence the ranking performance much, therefor it is not used in the item ranking task.

Performance Comparison. Ranking performance in terms of multiple standard ranking metrics is shown in Table 1. In particular, several observations can be made. Firstly, the proposed approach consistently achieves the best performance on all datasets, ascertaining the effectiveness of FML on item ranking. The overall performance gaining over WRMF model is up to 20.8%. Secondly, although both FML and CML rely on metric learning, our model still outperforms the CML sufficiently by up to 8.7%, which indicates that FML can reflect the distance relationship among users and items more accurately. Thirdly, Euclidean distance based models (FML and CML) perform better than dot product based recommender and neural network models, which is consistent with the results reported in [15]. This also verifies the superiority of using Euclidean distance to capture the interaction patterns. Fourthly, neural network based model CDAE is a strong baseline. It usually outperforms BPR and WRMF, while NeuMF seems to underperform than CDAE. However, there is still a huge performance gap

[4] http://research.yahoo.com/Academic_Relations
[5] http://www.cs.cmu.edu/ lebanon/IR-lab/data.html
[6] http://mymedialite.net/index.html
[7] https://www.librec.net/
[8] http://mymedialite.net/index.html
[9] https://github.com/hexiangnan/neural_collaborative_filtering

Table 1: Performance comparison on item ranking task in terms of MAP, MRR, NDCG, Precision and Recall on four datasets. Best performance is in boldface and second best is underlined.

Methods	MAP	MRR	NDCG	Precision@5	Precision@10	Recall@5	Recall@10
YahooMovie							
BPR	0.161 ± 0.001	0.322 ± 0.001	0.387 ± 0.001	0.128 ± 0.001	0.096 ± 0.002	0.161 ± 0.001	0.240 ± 0.001
NeuMF	0.171 ± 0.001	0.329 ± 0.006	0.402 ± 0.004	0.140 ± 0.002	0.112 ± 0.001	0.168 ± 0.002	0.256 ± 0.003
WRMF	0.209 ± 0.001	0.384 ± 0.002	0.435 ± 0.004	0.176 ± 0.001	0.135 ± 0.001	0.209 ± 0.001	0.310 ± 0.001
CDAE	0.224 ± 0.002	0.409 ± 0.005	0.453 ± 0.003	0.188 ± 0.003	0.143 ± 0.002	0.226 ± 0.003	0.331 ± 0.004
CML-PAIR	0.223 ± 0.002	0.416 ± 0.003	0.456 ± 0.003	0.189 ± 0.004	0.147 ± 0.002	0.224 ± 0.003	0.332 ± 0.002
CML-WARP	0.235 ± 0.002	0.440 ± 0.005	0.466 ± 0.001	0.202 ± 0.001	0.153 ± 0.001	0.237 ± 0.002	0.342 ± 0.002
FML	**0.262** ± 0.001	**0.466** ± 0.002	**0.486** ± 0.001	**0.222** ± 0.001	**0.167** ± 0.001	**0.262** ± 0.002	**0.374** ± 0.002
YahooMusic							
BPR	0.133 ± 0.001	0.300 ± 0.002	0.373 ± 0.001	0.118 ± 0.001	0.092 ± 0.001	0.141 ± 0.002	0.215 ± 0.001
NeuMF	0.141 ± 0.002	0.319 ± 0.005	0.383 ± 0.003	0.130 ± 0.002	0.099 ± 0.001	0.155 ± 0.001	0.232 ± 0.003
WRMF	0.144 ± 0.001	0.333 ± 0.003	0.386 ± 0.001	0.130 ± 0.002	0.097 ± 0.001	0.153 ± 0.002	0.226 ± 0.002
CDAE	0.153 ± 0.003	0.348 ± 0.006	0.395 ± 0.004	0.136 ± 0.002	0.102 ± 0.001	0.163 ± 0.002	0.240 ± 0.003
CML-PAIR	0.148 ± 0.002	0.339 ± 0.003	0.390 ± 0.003	0.134 ± 0.003	0.102 ± 0.002	0.159 ± 0.002	0.236 ± 0.004
CML-WARP	0.153 ± 0.002	0.353 ± 0.005	0.393 ± 0.002	0.140 ± 0.002	0.103 ± 0.001	0.171 ± 0.002	0.249 ± 0.003
FML	**0.169** ± 0.001	**0.375** ± 0.002	**0.411** ± 0.001	**0.152** ± 0.001	**0.114** ± 0.001	**0.183** ± 0.001	**0.268** ± 0.002
EachMovie							
BPR	0.394 ± 0.003	0.627 ± 0.002	0.641 ± 0.002	0.346 ± 0.001	0.293 ± 0.001	0.313 ± 0.001	0.448 ± 0.001
NeuMF	0.414 ± 0.003	0.656 ± 0.002	0.657 ± 0.003	0.378 ± 0.003	0.302 ± 0.002	0.335 ± 0.003	0.475 ± 0.001
WRMF	0.433 ± 0.001	0.679 ± 0.001	0.670 ± 0.002	0.397 ± 0.002	0.314 ± 0.001	0.355 ± 0.002	0.494 ± 0.001
CDAE	0.432 ± 0.003	0.678 ± 0.003	0.673 ± 0.002	0.394 ± 0.004	0.311 ± 0.003	0.356 ± 0.003	0.497 ± 0.002
CML-PAIR	0.426 ± 0.003	0.674 ± 0.002	0.668 ± 0.001	0.399 ± 0.001	0.315 ± 0.002	0.348 ± 0.002	0.492 ± 0.001
CML-WARP	0.419 ± 0.001	0.683 ± 0.001	0.663 ± 0.002	0.398 ± 0.001	0.312 ± 0.001	0.346 ± 0.002	0.485 ± 0.003
FML	**0.454** ± 0.001	**0.696** ± 0.002	**0.687** ± 0.001	**0.416** ± 0.001	**0.330** ± 0.001	**0.370** ± 0.002	**0.515** ± 0.002
FilmTrust							
BPR	0.476 ± 0.004	0.600 ± 0.007	0.635 ± 0.003	0.412 ± 0.005	0.347 ± 0.001	0.391 ± 0.009	0.613 ± 0.007
NeuMF	0.483 ± 0.001	0.609 ± 0.005	0.646 ± 0.003	0.413 ± 0.003	0.350 ± 0.002	0.393 ± 0.004	0.626 ± 0.007
WRMF	0.516 ± 0.002	0.648 ± 0.005	0.663 ± 0.002	0.433 ± 0.002	0.351 ± 0.001	0.427 ± 0.005	0.632 ± 0.007
CDAE	0.523 ± 0.008	0.654 ± 0.010	0.678 ± 0.008	0.436 ± 0.004	0.353 ± 0.003	0.441 ± 0.006	0.647 ± 0.005
CML-PAIR	0.491 ± 0.002	0.637 ± 0.003	0.655 ± 0.001	0.418 ± 0.002	0.337 ± 0.001	0.408 ± 0.003	0.602 ± 0.003
CML-WARP	0.529 ± 0.004	0.666 ± 0.005	0.684 ± 0.003	0.438 ± 0.005	0.359 ± 0.003	0.441 ± 0.007	0.653 ± 0.004
FML	**0.549** ± 0.005	**0.685** ± 0.006	**0.701** ± 0.004	**0.453** ± 0.003	**0.366** ± 0.002	**0.462** ± 0.006	**0.672** ± 0.006

between these models and FML, which also suggests that adding nonlinearity be not enough.

Overall, these observations demonstrate the advantages of considering the recommendation problem from the metric vector space factorization perspective. This also clearly answers the **RQ1**.

4.2 Evaluation for Rating Prediction

Datasets Descriptions. We evaluate the accuracy of FML on rating estimation using the following datasets:

- **Goodbooks**. This dataset contains six million ratings for about ten thousand books. It is collected from a real-word book website[10].
- **Movielens Datasets**. We use three versions of publicly available Movielens dataset including Movielens 100K,Movielens 1M and Movielens Latest. All of them are collected by GroupLens research[11], and Movielens Latest is a comparatively new dataset including ratings generated in the year 2016.

Evaluation Metrics. We employ two widely used metrics: Root Mean Squared Error (RMSE) and Mean Average Error (MAE), to assess prediction errors. MAE measures the average magnitude of errors in a set of predictions, while RMSE tends to disproportionately penalize large errors[27]. Since these metrics are commonplace for recommendation evaluation, we omit the details of evaluation metrics for brevity.

Baselines. We compare FML with the following baselines.

[10] http://fastml.com/goodbooks-10k

[11] https://grouplens.org/datasets/movielens/

Table 2: Comparison between our approach and baselines in terms of RMSE and MAE. Best performance is in boldface and second best is underlined.

Datasets	Goodbooks	Movielens Latest	Movielens 100K	Movielens 1M
Models	**Root Mean Squared Error**			
UserAverage	0.893 ± 0.001	0.967 ± 0.005	1.047 ± 0.004	1.036 ± 0.002
ItemAverage	0.953 ± 0.002	0.991 ± 0.008	1.035 ± 0.005	0.978 ± 0.001
SlopOne	0.825 ± 0.001	0.907 ± 0.007	0.935 ± 0.001	0.899 ± 0.002
BPMF	0.829 ± 0.001	0.932 ± 0.001	0.927 ± 0.003	0.867 ± 0.002
BiasedSVD	0.846 ± 0.001	0.868 ± 0.005	0.911 ± 0.003	0.847 ± 0.002
NRR	0.822 ± 0.001	0.876 ± 0.002	0.909 ± 0.003	0.875 ± 0.002
NNMF	0.821 ± 0.001	0.871 ± 0.006	0.903 ± 0.002	0.843 ± 0.002
FML	**0.799** ± 0.001	**0.853** ± 0.006	**0.891** ± 0.005	**0.836** ± 0.001
Models	**Mean Average Error**			
UserAverage	0.700 ± 0.001	0.753 ± 0.002	0.839 ± 0.004	0.830 ± 0.002
ItemAverage	0.762 ± 0.002	0.769 ± 0.004	0.825 ± 0.005	0.782 ± 0.001
SlopOne	0.638 ± 0.001	0.693 ± 0.007	0.737 ± 0.001	0.710 ± 0.002
BPMF	0.639 ± 0.001	0.707 ± 0.001	0.725 ± 0.003	0.678 ± 0.001
BiasedSVD	0.668 ± 0.001	0.666 ± 0.002	0.718 ± 0.002	0.665 ± 0.003
NRR	0.631 ± 0.003	0.674 ± 0.005	0.717 ± 0.005	0.691 ± 0.002
NNMF	0.637 ± 0.001	0.667 ± 0.004	0.709 ± 0.002	0.669 ± 0.001
FML	**0.607** ± 0.001	**0.656** ± 0.004	**0.696** ± 0.005	**0.654** ± 0.001

- **Average**. This approach predicts ratings based on average historical ratings and has two variants: UserAverage and ItemAverage.
- **SlopOne** [20]. It is an efficient online rating-based collaborative filtering approach.
- **BPMF** [29]. This model considers the matrix factorization from a probabilistic perspective and provides a fully Bayesian treatment by solving it with Markov chain Monte Carlo.
- **BiasedSVD** [19]. This model introduces the bias terms to both users and items. It equips vanilla matrix factorization to model the user/item specific characteristics.
- **NRR** [21]. NRR is a neural rating regression model which captures the user and item relationships with neural network regression.
- **NNMF** [5]. This model replaces the simple dot product of matrix factorization with multi-layered neural network to learn an arbitrary function for modeling user-item relationships.

Implementation Details. We randomly split each dataset into a training set and a test set by the ratio of 9:1, and report the average results over five different splits. Hyper-parameters for all models are tuned to achieve their best performance. Learning rate for FML is set to 0.05. $g(\cdot)$ is set to square function for Goodbooks, Movielens latest and 1M, and absolute function for Movielens 100k. Other hyper-parameters are set as follows: Goodbooks($\lambda = 0.01$, $\alpha = 0.2$, $k = 200$,$dropout = 0.1$, $\tau = 0.8$, and $l = 1.0$); Movielens Latest($\lambda = 0.05$, $\alpha = 0.2$, $k = 250$, $dropout = 0.15$, $\tau = 0.8$, and $l = 0.85$); Movielens 100K($\lambda = 0.01$, $\alpha = 0.2$, $k = 150$, $dropout = 0.05$, $\tau = 0.9$, and $l = 1.0$); Movielens 1M($\lambda = 0.01$, $\alpha = 0.1$, $k = 150$, $dropout = 0.03$, $\tau = 0.5$, and $l = 1.4$).

Performance Comparison. From Table 2, we observe that our model consistently outperforms all compared methods including matrix factorization based models as well as neural network based models. Specifically, FML achieves substantial improvement over BPMF and BiasedSVD. Naive approaches UserAverage and ItemAverage perform poorly on all datasets. Recent neural network based model NRR and NNMF are two competitive baselines especially on large scale dataset Goodbooks. However, NRR does not perform well on other three datasets. With this table, we can give a sure answer to **RQ2**.

4.3 MODEL ANALYSIS AND DISCUSSION

In this section, we study the impact of key hyper-parameter settings on FML for both item ranking and rating prediction, to answer **RQ3**. All the experiments are performed on FilmTrust (Item Ranking) and Movielens Latest (Rating Prediction). Due to limited space, we only report two metrics (RMSE and NDCG) and two baselines (WRMF and BiasedSVD).

Number of Dimension k. Figure 2 (a) shows the rating prediction performance varied across different k. We observe that our model consistently outperforms BiasedSVD, and it reaches its peak with a smaller k value. Figure 3 (a) shows the effect of varying k on item ranking. Clearly, WRMF is prone to overfitting for larger size of dimensions. Its best performance is usually achieved with a comparatively small k, then it decreases rapidly (similar phenomenon is also observed on other datasets, e.g, it starts to overfit when we set k to 50 on YahooMovie.). This might be caused by the weakness of dot product and obviously limits its the flexibility and expressiveness. However, our model FML is demonstrating consistent stability, and less likely to overfit with comparatively large number of latent

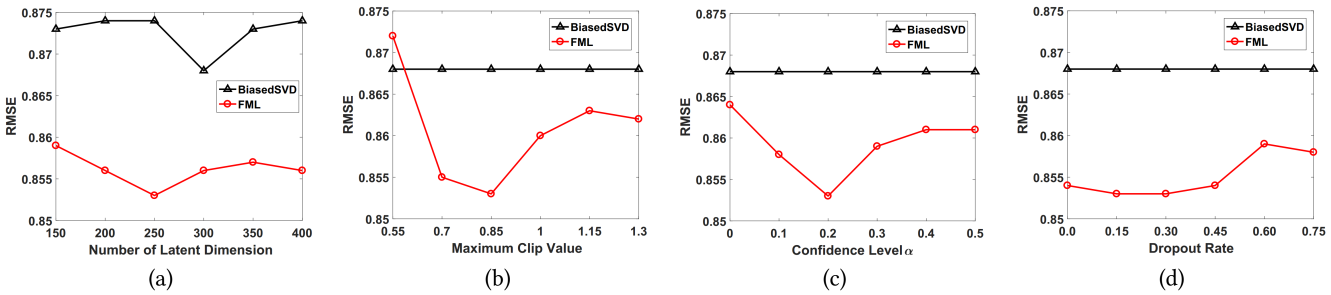


Figure 2: Effect of Hyper-parameter settings on FML for rating prediction on Movielens Latest (The lower the better).

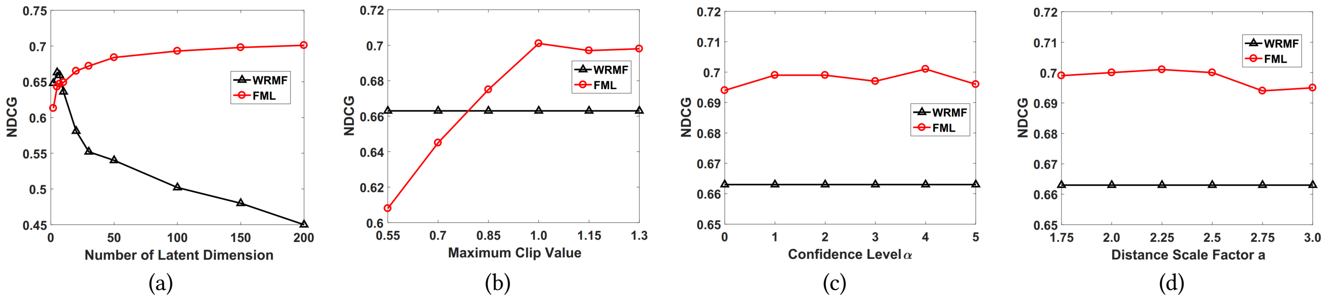


Figure 3: Effect of Hyper-parameter settings on FML for item ranking on FilmTrust (The higher the better).

dimension. Furthermore, the performance is usually enhanced by increasing the dimension size, allowing it to capture more factors of user preferences and item characteristics on personalized ranking task.

Maximum Clip Value l. Figure 2 (b) and 3 (b) shows the effect of maximum clip value l on model performance. l is a critical model parameters as it controls the scope of user and item positions. Intuitively, ideal clip value might be determined by the range of feedback values (e.g., ratings or implicit feedback). For rating estimation, it is sensible to tune l carefully. For item ranking task, since the implicit feedback is either 0 or 1, setting l to 1 is sufficient.

Confidence Level α. Here, we investigate the impact of hyper-parameter α. Setting $\alpha = 0$ means that confidence is removed. For rating prediction, α works with function $g(\cdot)$ to reduce the potential noise in rating data and increase the model robustness. For item ranking, α scales the implicit feedback to differentiate between positive samples and negative samples. As shown in Figure 2 (c) and 3 (c), using confidence does lead to improved performance on both cases. A practical guideline for choosing α is to determine it based on how much confidence you have about your estimation.

Dropout Rate. Figure 2(d) shows the test performance varied across different dropout rate. It is observed that dropout does improve the performance of rating prediction. Nonetheless, setting the dropout rate to a small value is more sensible as it can retain more information. Unexpectedly, we found that dropout operation does not improve the performance of ranking based FML.

Distance Scale Factor a. This hyper-parameter controls the minimum distance that we would like to push the negative items away from the users. As shown in Figure 3(d), a does not have as much influence as that of other parameters. Obviously, the ranking performance on FilmTrust does not fluctuate much when we change a from 1.75 to 2.5.

Visualization of Item Embedding. Figure 4 illustrates the T-SNE [23] embeddings of FML and the CML-WARP. Color of each point indicates the genre of movie. Note that each movie may have many genres and we randomly select one genre as the label. The distribution of genres is quite random since neither FML nor CML are trained for genre classification. We observe that the item embeddings tend to formulate some clusters even though no side information is given during learning. This might indicate some latent correlations of items. It is obvious that FML get two clusters with more clear boundary, while the gap between the two clusters in CML is somehow congested. This phenomena explains the reason why FML achieves better performance than CML. Through direct factorization, FML manages to overcome the over-congestion issue of CML.

5 CONCLUSION AND FUTURE WORK

In this paper, we propose FML which treats users and items as points in a metric vector space. We learn the positions of these points via factorization, which enables FML to address the two classic recommendation tasks: rating prediction and item ranking. The idea is intuitive yet effective with high flexibility. Experiments on a number of real world datasets from different domains showed that our model outperforms the state-of-the-art.

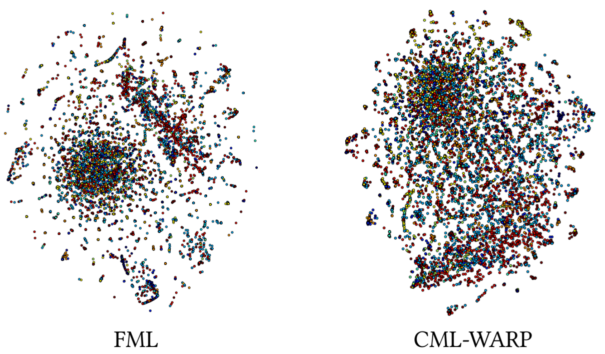


Figure 4: T-SNE visualization of item embeddings on YahooMovie. Left: FML; Right: CML-WARP.

For future work, we will explore its extensions by accommodating temporal dynamics and multi-dimensional side information such as item and user content descriptions, review texts and trust/social relationships. Furthermore, this model might be useful in network embedding [7], i.e., to factorize the ***distance*** between vertex to learn vertex representations.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin. 2005. 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 (June 2005), 734–749. https://doi.org/10.1109/TKDE.2005.99

[2] Xavier Amatriain, Josep M. Pujol, and Nuria Oliver. 2009. I Like It... I Like It Not: Evaluating User Ratings Noise in Recommender Systems. In *Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization: Formerly UM and AH (UMAP '09)*. Springer-Verlag, Berlin, Heidelberg, 247–258.

[3] Ivan Dokmanic, Reza Parhizkar, Juri Ranieri, and Martin Vetterli. 2015. Euclidean distance matrices: essential theory, algorithms, and applications. *IEEE Signal Processing Magazine* 32, 6 (2015), 12–30.

[4] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12, Jul (2011), 2121–2159.

[5] Gintare Karolina Dziugaite and Daniel M. Roy. 2015. Neural Network Matrix Factorization. *CoRR* abs/1511.06443 (2015). arXiv:1511.06443 http://arxiv.org/abs/1511.06443

[6] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2001. *The elements of statistical learning*. Vol. 1. Springer series in statistics New York.

[7] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[8] Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. 2009. Is that you? Metric learning approaches for face identification. In *ICCV 2009-International Conference on Computer Vision*. IEEE, 498–505.

[9] G. Guo, J. Zhang, and N. Yorke-Smith. 2013. A Novel Bayesian Similarity Measure for Recommender Systems. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*. 2619–2625.

[10] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 507–517. https://doi.org/10.1145/2872427.2883037

[11] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI'16)*. AAAI Press, 144–150. http://dl.acm.org/citation.cfm?id=3015812.3015834

[12] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. ACM, New York, NY, USA, 355–364. https://doi.org/10.1145/3077136.3080777

[13] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW 2017*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 173–182. https://doi.org/10.1145/3038912.3052569

[14] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*. 549–558. https://doi.org/10.1145/2911451.2911489

[15] Cheng-Kang Hsieh, Longqi Yang, Yin Cui, Tsung-Yi Lin, Serge Belongie, and Deborah Estrin. 2017. Collaborative Metric Learning. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 193–201. https://doi.org/10.1145/3038912.3052639

[16] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM*. IEEE Computer Society, Washington, DC, USA, 263–272. https://doi.org/10.1109/ICDM.2008.22

[17] Nicolas Jones, Armelle Brun, and Anne Boyer. 2011. Comparisons Instead of Ratings: Towards More Stable Preferences. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01 (WI-IAT '11)*. IEEE Computer Society, Washington, DC, USA, 451–456. https://doi.org/10.1109/WI-IAT.2011.13

[18] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08)*. ACM, New York, NY, USA, 426–434. https://doi.org/10.1145/1401890.1401944

[19] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (Aug. 2009), 30–37. https://doi.org/10.1109/MC.2009.263

[20] Daniel Lemire and Anna Maclachlan. 2005. Slope one predictors for online rating-based collaborative filtering. In *Proceedings of the 2005 SIAM International Conference on Data Mining*. SIAM, 471–475.

[21] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural Rating Regression with Abstractive Tips Generation for Recommendation. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '17)*. ACM, New York, NY, USA, 345–354. https://doi.org/10.1145/3077136.3080822

[22] Shengcai Liao, Yang Hu, Xiangyu Zhu, and Stan Z Li. 2015. Person re-identification by local maximal occurrence representation and metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2197–2206.

[23] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[24] Aäron van den Oord, Sander Dieleman, and Benjamin Schrauwen. 2013. Deep Content-based Music Recommendation. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 2643–2651. http://dl.acm.org/citation.cfm?id=2999792.2999907

[25] Parikshit Ram and Alexander G. Gray. 2012. Maximum Inner-product Search Using Cone Trees. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12)*. ACM, New York, NY, USA, 931–939. https://doi.org/10.1145/2339530.2339677

[26] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09)*. AUAI Press, Arlington, Virginia, United States, 452–461.

[27] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. 2010. *Recommender Systems Handbook* (1st ed.). Springer-Verlag New York, Inc., New York, NY, USA.

[28] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS'07)*. Curran Associates Inc., USA, 1257–1264.

[29] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th international conference on Machine learning*. ACM, 880–887.

[30] Guy Shani and Asela Gunawardana. 2011. Evaluating recommendation systems. *Recommender systems handbook* (2011), 257–297.

[31] Guy Shani and Asela Gunawardana. 2011. *Evaluating Recommendation Systems*. Springer US, Boston, MA, 257–297. https://doi.org/10.1007/978-0-387-85820-3_8

[32] Yue Shi, Martha Larson, and Alan Hanjalic. 2014. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)* 47, 1 (2014), 3.

[33] Yue Shi, Martha Larson, and Alan Hanjalic. 2014. Collaborative Filtering Beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges. *ACM Comput. Surv.* 47, 1, Article 3 (May 2014), 45 pages. https://doi.org/10.1145/2556270

[34] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[35] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018. CoupleNet: Paying Attention to Couples with Coupled Attention for Relationship Recommendation. *CoRR*

abs/1805.11535 (2018). arXiv:1805.11535 http://arxiv.org/abs/1805.11535

[36] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018. Multi-Pointer Co-Attention Networks for Recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &#38; Data Mining (KDD '18)*. ACM, New York, NY, USA, 2309–2318. https://doi.org/10.1145/3219819.3220086

[37] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. Latent Relational Metric Learning via Memory-based Attention for Collaborative Ranking. In *WWW 2018, Lyon, France, April 23-27, 2018*. 729–739. https://doi.org/10.1145/3178876.3186154

[38] Amos Tversky and Itamar Gati. 1982. Similarity, separability, and the triangle inequality. *Psychological review* 89, 2 (1982), 123.

[39] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. 2015. Collaborative Deep Learning for Recommender Systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15)*. ACM, New York, NY, USA, 1235–1244. https://doi.org/10.1145/2783258.2783273

[40] Kilian Q. Weinberger and Lawrence K. Saul. 2009. Distance Metric Learning for Large Margin Nearest Neighbor Classification. *J. Mach. Learn. Res.* 10 (June 2009), 207–244.

[41] Jason Weston, Samy Bengio, and Nicolas Usunier. 2010. Large scale image annotation: learningÂătoÂărank withÂăjoint word-image embeddings. *Machine Learning* 81, 1 (01 Oct 2010), 21–35. https://doi.org/10.1007/s10994-010-5198-3

[42] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. 2016. Collaborative Denoising Auto-Encoders for Top-N Recommender Systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining (WSDM '16)*. ACM, New York, NY, USA, 153–162. https://doi.org/10.1145/2835776.2835837

[43] Eric P Xing, Michael I Jordan, Stuart J Russell, and Andrew Y Ng. 2003. Distance metric learning with application to clustering with side-information. In *Advances in neural information processing systems*. 521–528.

[44] Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep Learning based Recommender System: A Survey and New Perspectives. *arXiv preprint arXiv:1707.07435* (2017).

[45] Shuai Zhang, Lina Yao, Aixin Sun, Sen Wang, Guodong Long, and Manqing Dong. 2018. NeuRec: On Nonlinear Transformation for Personalized Ranking. *arXiv preprint arXiv:1805.03002* (2018).

[46] Shuai Zhang, Lina Yao, and Xiwei Xu. 2017. AutoSVD++: An Efficient Hybrid Collaborative Filtering Model via Contractive Auto-encoders. In *SIGIR*. ACM, New York, NY, USA, 957–960. https://doi.org/10.1145/3077136.3080689

[47] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W. Bruce Croft. 2017. Joint Representation Learning for Top-N Recommendation with Heterogeneous Information Sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM '17)*. ACM, New York, NY, USA, 1449–1458. https://doi.org/10.1145/3132847.3132892