2018 IEEE International Conference on Data Mining

Collaborative Translational Metric Learning

Chanyoung Park[1], Donghyun Kim[2], Xing Xie[3], Hwanjo Yu[1*]
Dept. of Computer Science and Engineering, POSTECH, South Korea[1]
Oath, USA[2]
Microsoft Research Asia, China[3]
{pcy1302, hwanjoyu}@postech.ac.kr, cartopy@gmail.com, xingx@microsoft.com

***Abstract*—Recently, matrix factorization–based recommendation methods have been criticized for the problem raised by the triangle inequality violation. Although several metric learning–based approaches have been proposed to overcome this issue, existing approaches typically project each user to a single point in the metric space, and thus do not suffice for properly modeling the *intensity* and the *heterogeneity* of user–item relationships in implicit feedback. In this paper, we propose TransCF to discover such latent user–item relationships embodied in implicit user–item interactions. Inspired by the translation mechanism popularized by knowledge graph embedding, we construct user–item specific translation vectors by employing the neighborhood information of users and items, and translate each user toward items according to the user's relationships with the items. Our proposed method outperforms several state-of-the-art methods for top-N recommendation on seven real-world data by up to 17% in terms of hit ratio. We also conduct extensive qualitative evaluations on the translation vectors learned by our proposed method to ascertain the benefit of adopting the translation mechanism for implicit feedback–based recommendations.**

***Index Terms*—Recommender system, Metric learning, Collaborative filtering**

I. Introduction

The recent explosive growth of information on the Internet inundates users with choices, and recommender systems play a crucial role not only in helping users in their decision making, but also in increasing the revenues of e-commerce companies. Among various recommendation techniques, matrix factorization (MF)–based collaborative filtering (CF) [1] has been shown to be the most successful; it assumes that users who have had similar interests in the past will tend to share similar interests in the future [2]. More specifically, MF learns low-rank vector representations of users and items from their previous interaction history and models the similarity of user–item pairs using inner products. However, a critical yet not widely recognized flaw of employing the inner product as a similarity metric is that it violates the triangle inequality [3]. As a concrete example, if user u_1 liked items v_1 and v_2, MF will put both items close to u_1, but will not necessarily place v_1 and v_2 close to each other. That is to say, the seemingly obvious item–item similarity (between v_1 and v_2) is not guaranteed to be learned when the triangle inequality is violated [4].

To address the above limitation of MF–based recommendation, several metric learning approaches have been proposed [4]–[7]. They commonly project entities (users and items) into a low-dimensional metric space, i.e., Euclidean space, where the similarity between points is inversely proportional to the Euclidean distance that satisfies the triangle inequality. Specifically, CML [4] is the state-of-the-art metric learning–based recommendation method for implicit feedback (e.g., clicks or purchases); it minimizes the distances between embeddings of a user and items that the user has interacted with, under the assumption that a user should be closer to the items the user likes than to those that the user does not. In this way, these approaches not only expect to propagate positive user–item relationships to other unknown user–item pairs, but also to capture the similarity within user–user and item–item pairs by satisfying the triangle inequality.

Although existing metric learning approaches have shown their effectiveness by satisfying the triangle inequality, they suffer from an inherent limitation, which is that *each user is projected to a single point in the metric space*. Such a rigid restriction imposed on user embeddings makes it hard to properly model the **intensity** and the **heterogeneity** of user–item relationships in implicit feedback. More precisely, a user's implicit feedback on multiple items does not necessarily indicate that he holds an equal preference for these items; rather, some of the items are more relevant to the user than others. This implies that the *intensity* of the user–item relationships embodied in implicit user–item interactions differ from one another. Moreover, a user may have a wide variety of tastes in different item categories, implying that the type of user–item relationship is *heterogeneous* with regard to the user's tastes in various item categories[1]. However, it is by no means an easy task to project each user to a single point such that his intense and heterogeneous relationships with items are fully considered, and this phenomenon compounds as the number of users and items increases.

This paper presents a novel method called TransCF to overcome the above limitation of existing metric learning approaches. We achieve this by adopting the *translation mechanism*, which has been proven to be effective for knowledge graph embedding [8], [9], by which the relations between entities are interpreted as translation operations between them. In our work, we embed users and items as points in a low-dimensional metric space, and additionally introduce translation vectors to translate each user to multiple points. Equipped with the user–item specific translation vectors, a user is translated toward his relevant items by considering his relationships with the items, which facilitates the modeling of the intensity and the heterogeneity of user–item relationships in implicit feedback that were overlooked by the previous

*Corresponding Author

[1]In Section II, we show the existence of the intensity and the heterogeneity of user–item relationships in implicit feedback.

978-1-5386-9159-5/18/$31.00 ©2018 IEEE
DOI 10.1109/ICDM.2018.00052

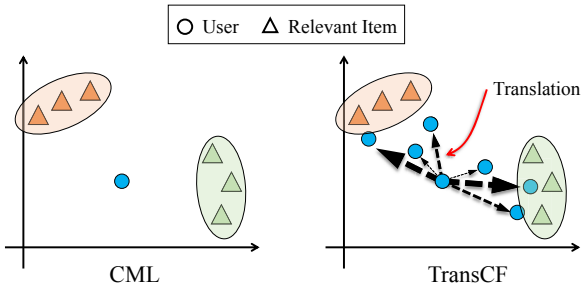
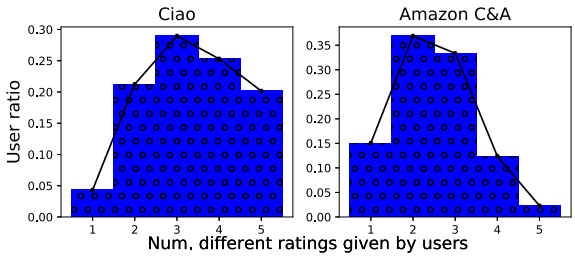


Fig. 1: Toy example illustrating the benefit of the user–item specific translation vectors. Items are assumed to be grouped according to their categories.

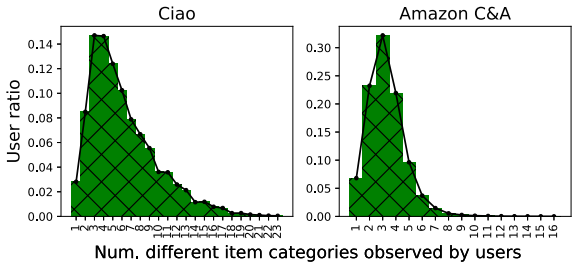
metric learning approaches. A further appeal of our proposed method is the ability to handle the *complex* nature of CF where it is common for a user to interact with multiple items, i.e., one-to-many mapping. Whereas it is not feasible for the previous metric learning approaches to pull a user closer to all of the items (one-to-many mapping) because a user is projected to a single point, our proposed method alleviates this issue because once a user is translated to multiple points, a one-to-many mapping can be deemed as multiple one-to-one mappings. As an illustration, consider the following toy example.

Toy Example. In Figure 1, whereas CML tries to find a single point for a user that minimizes the distances between the user and his relevant items, TransCF introduces user–item specific translation vectors to translate the user to multiple points according to the intensity and the heterogeneity of the user–item relationships. The more intense (thickness of the vectors) the user–item relationship, the closer the user is expected to be translated to the item. Besides, the direction of the vectors and the angles between them reflect the heterogeneity of the relationship with regard to the user's tastes in various item categories.

However, directly applying the translation mechanism into a general recommendation framework for implicit feedback is not viable because *user–item interactions in the implicit feedback dataset are not labeled*, unlike in knowledge graphs where each relation between entities is given a label, such as "was_born_in" or "performed_in". Therefore, the key for successfully adopting the translation mechanism in our framework boils down to defining labels for implicit user–item interactions, and materializing them into the user–item specific translation vectors. Inspired by the highly effective neighborhood–based CF algorithms [1], [10]–[14], whose assumptions are that similar users display similar item preferences and that similar items are consumed by similar users, we employ the neighborhood information of users and items to construct the translation vectors (**Section III-C**). Combined with a regularizer, specifically tailored to TransCF, that guides each user/item to its neighbors (**Section III-D1**), TransCF explicitly models the neighborhood information, in contrast to CML, which expects to achieve it implicitly by satisfying the triangle inequality. It is worth noting that the user–item specific translation vectors are constructed without introducing any new parameters, which prevents our proposed method from overfitting. Furthermore, we introduce a second regularizer to further improve the accuracy of TransCF in case the user–item relationships become more complex (**Section III-D2**). Our extensive experiments on seven real-world datasets (**Section IV-A**) show that TransCF considerably outperforms the state-of-the-art recommendation methods for implicit feedback by up to 17% in terms of hit ratio. We also perform various experiments to qualitatively ascertain that TransCF indeed benefit from modeling the intensity and the heterogeneity of user–item relationships in implicit feedback as illustrated in Figure 1 (**Section IV-B**). The source code is available at https://github.com/pcy1302/TransCF.



(a) Distribution of the number of different ratings given by users.



(b) Dist. of the num. different item categories observed by users.

Fig. 2: Data analysis on Ciao and Amazon C&A datasets.

II. Data Analysis: Intensity and Heterogeneity

In this section, we perform data analyses on Ciao [15] and Amazon C&A [16] datasets to provide evidences of the existence of the *intensity* and the *heterogeneity* of user–item relationships in implicit feedback. The ratings in both datasets are integers from 1 to 5, and the numbers of unique item categories are 28 and 45, respectively. In our experiments, we regard every user–item interaction with a rating as an implicit feedback record[2]. Figure 2 shows the distribution of the number of different ratings and different item categories given and observed by users. In Figure 2a, we observe that most users (95% in Ciao and 85% in Amazon C&A) give two or more different ratings. This implies that implicit user–item interactions do indeed encode different **intensities** of user–item relationships, assuming that the rating is a proxy for the intensity; a higher rating implies higher intensity. Moreover, in Figure 2b we observe that only a few users (3% in Ciao and 6% in Amazon C&A) interact with a single category,

[2]It is unavoidable to use explicit feedback datasets as it is not feasible to judge the intensity of user–item relationships from implicit feedback.

whereas the vast majority of users interact with two or more different item categories. This implies that users have diverse tastes in various item categories, and thus the type of user–item relationship is **heterogeneous** with regard to the users' tastes in various item categories.

In Section IV-B2, we show by our experiments that TransCF can inversely infer knowledge regarding the intensity and the heterogeneity from the implicit user–item interactions.

III. Proposed Method: TransCF

In this section, we give details of our proposed method, which adopts the translation mechanism for modeling the intensity and the heterogeneity of user–item relationships in implicit feedback.

A. Problem Formulation

In this work, we focus on recommendations for implicit feedback. Let $\mathcal{U}$ and $\mathcal{I}$ denote a set of users and a set of items, respectively. $\mathcal{N}_u^{\mathcal{I}}$ denotes the set of items that user u has previously interacted with. Given implicit user–item interactions (e.g., bookmarks and purchase history), our goal is to recommend items $i \in \mathcal{I} \backslash \mathcal{N}_u^{\mathcal{I}}$ to each user $u \in \mathcal{U}$. Note that we use neither the rating information nor any auxiliary information regarding users and items (e.g., user social network or item category).

B. Scoring Function

Recall that the objectives of this work are 1) to address the limitation of the inner product as a scoring function, which violates the triangle inequality, and 2) to model the intensity and the heterogeneity of user–item relationships in implicit feedback. To this end, we adopt Euclidean distance as our distance metric to satisfy the triangle inequality, and we introduce translation vectors to model the intensity and the heterogeneity of implicit user–item interactions. Given K-dimensional embedding vectors $\boldsymbol{\alpha}_u \in \mathbb{R}^K$ for user u, $\boldsymbol{\beta}_i \in \mathbb{R}^K$ for item i, and $\boldsymbol{r}_{ui} \in \mathbb{R}^K$ for the translation of user u with regard to item i, the similarity score $s(u,i)$ between user u and item i is:

$$s(u,i) = -\left\|\boldsymbol{\alpha}_u + \boldsymbol{r}_{ui} - \boldsymbol{\beta}_i\right\|_2^2 \tag{1}$$

where a higher similarity score $s(u,i)$ implies a higher probability that user u will like item i. In other words, the similarity score between user u and item i is computed by the distance between the translated embedding vector of user u, given by $(\boldsymbol{\alpha}_u + \boldsymbol{r}_{ui})$, and the embedding vector $\boldsymbol{\beta}_i$ of item i.

Optimization Objective. Given the scoring function $s(u,i)$ as in Equation 1, we minimize a popular margin–based pairwise ranking criterion [4], [8], [9], i.e., hinge loss, as follows:

$$\mathcal{L}(\Theta) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{N}_u^{\mathcal{I}}} \sum_{j \notin \mathcal{N}_u^{\mathcal{I}}} [\gamma - s(u,i) + s(u,j)]_+ \tag{2}$$

where $[x]_+ = max(x, 0)$, and γ is the margin. Our objective is to ensure that the similarity score of an observed (positive) user–item pair (u,i) is higher than that of an unobserved (negative) pair (u,j) by a margin of at least γ. By doing so, we aim to translate each user u closer to his relevant item i while translating him farther away from his non-relevant item j. One thing to note is that the translation vector $\boldsymbol{r}_{ui}$ is user–item specific: *it translates user u with respect to item i according to the user's specific relationship with the item i.* This property enables TransCF not only to capture the intensity and the heterogeneity of user–item relationships in implicit feedback, but also to handle the complex nature of CF.

C. Modeling Relations: Neighborhood Information

As mentioned previously, the key for successfully adopting the translation mechanism in our framework is modeling the translation vectors so that they reflect the intensity and the heterogeneity of user–item relationships in implicit feedback. The translation vectors can be flexibly constructed as long as the user–item interactions are properly modeled. Although it is possible to introduce a new parameter for each translation vector corresponding to every user–item pair, we note that not only is this approach prone to overfitting owing to the large number of parameters, but also it does not allow the collaborative information to be explicitly modeled as no parameters are shared among users and among items. Therefore, in this work we employ the neighborhood information of users and items to construct the translation vectors without introducing any new parameters. Neighborhood information, which has been shown to be highly effective for recommendation [11]–[14], implies that users are represented through the items that they prefer [1], [10], and items are represented through the users that prefer them. More precisely, considering that user embedding vectors reveal users' tastes, each item can be regarded as the average tastes of the users that have interacted with that item:

$$\boldsymbol{\beta}_i^{\text{nbr}} = \frac{1}{|\mathcal{N}_i^{\mathcal{U}}|} \sum_{k \in \mathcal{N}_i^{\mathcal{U}}} \boldsymbol{\alpha}_k \tag{3}$$

where $\boldsymbol{\beta}_i^{\text{nbr}} \in \mathbb{R}^K$ denotes the neighborhood embedding of item i, and $\mathcal{N}_i^{\mathcal{U}}$ denotes the set of users that have previously interacted with item i. Likewise, considering that item embedding vectors capture the prominent features of items, e.g., the item category, each user's taste can be represented by the average features of the items that the user has interacted with:

$$\boldsymbol{\alpha}_u^{\text{nbr}} = \frac{1}{|\mathcal{N}_u^{\mathcal{I}}|} \sum_{k \in \mathcal{N}_u^{\mathcal{I}}} \boldsymbol{\beta}_k \tag{4}$$

where $\boldsymbol{\alpha}_u^{\text{nbr}} \in \mathbb{R}^K$ denotes the neighborhood embedding of user u, and $\mathcal{N}_u^{\mathcal{I}}$ denotes the set of items that user u has previously interacted with. Given the respective representations of an item and a user from the neighborhood perspective as in Equations 3 and 4, we use them to model the user–item interactions as follows:

$$\boldsymbol{r}_{ui} = f(\boldsymbol{\alpha}_u^{\text{nbr}}, \boldsymbol{\beta}_i^{\text{nbr}})$$

where $f(x,y)$ denotes a function to model the interaction between the two input vectors x and y. Note that although we could employ various functions such as a multi-layer perceptron, it turns out that a simple element-wise vector product provides the highest accuracy.

Figure 3 illustrates a working example of TransCF. Given that user 1 has interacted with items $\{2,5,8\}$ and item 2 has been interacted with by users $\{1,3,6\}$, we obtain the

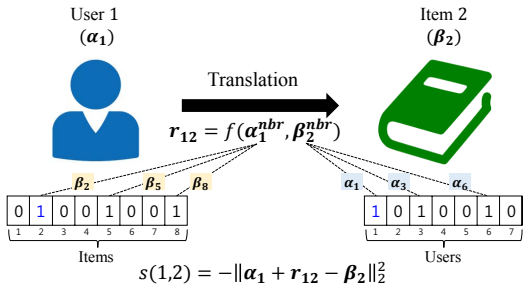


Fig. 3: An overview of TransCF.

neighborhood embedding vector $\boldsymbol{\alpha}_1^{\text{nbr}}$ of user 1 from the set of items $\{2, 5, 8\}$, and the neighborhood embedding vector $\boldsymbol{\beta}_2^{\text{nbr}}$ of item 2 from the set of users $\{1, 3, 6\}$. Then, using these neighborhood embedding vectors $\boldsymbol{\alpha}_1^{\text{nbr}}$ and $\boldsymbol{\beta}_2^{\text{nbr}}$, we construct the translation embedding vector $\boldsymbol{r}_{12} = f(\boldsymbol{\alpha}_1^{\text{nbr}}, \boldsymbol{\beta}_2^{\text{nbr}})$, which is eventually used for computing the similarity score $s(1,2) = -\|\boldsymbol{\alpha}_1 + \boldsymbol{r}_{12} - \boldsymbol{\beta}_2\|_2^2$.

*1) **Discussion**:* If users' social network information or auxiliary information related to items is given, it would be more intuitive to represent a user by his friends' tastes, and an item by its semantically related items, e.g., items that belong to the same category. However, such side information on users and items is not always provided in practice, and thus we do not consider any side information in our current work. Additionally, instead of simply averaging the embeddings of neighbors as in Equations 3 and 4, we could expect further improvements by applying the attention mechanism [17], whereby we give higher weights to more influential neighbors. Lastly, we could try swapping the role of users and items such that we conversely translate items towards users. However, since the above extensions are straightforward and our main focus is to incorporate the translation mechanism into recommender systems, we leave these for future studies.

D. Model Regularization

In this section, we introduce two regularizers that are tailored to TransCF. We later show in our experiments (Section IV-A2) that these regularizers are indeed beneficial.

*1) **Regularizer 1 – Neighborhood Regularizer**:* In Section III-C, we explained how we reflect the neighborhood information of users and items into our translation vectors, under the assumption that users and items can be represented by their neighbors. In the case of users, we implicitly assumed that $\boldsymbol{\alpha}_u$ can be represented by $\boldsymbol{\alpha}_u^{\text{nbr}}$ (Equation 4), and likewise for items, that $\boldsymbol{\beta}_i$ can be represented by $\boldsymbol{\beta}_i^{nbr}$ (Equation 3). However, to ensure that the neighborhood information is explicitly incorporated into our model, we need to guide $\boldsymbol{\alpha}_u$ to be close to $\boldsymbol{\alpha}_u^{\text{nbr}}$, and $\boldsymbol{\beta}_i$ to be close to $\boldsymbol{\beta}_i^{\text{nbr}}$. To this end, we introduce a regularizer named $reg_{\text{nbr}}(\Theta)$ that minimizes the distance between users/items and their neighbors:

$$reg_{\text{nbr}}(\Theta) = \sum_{u \in \mathcal{U}} \left(\boldsymbol{\alpha}_u - \frac{1}{|\mathcal{N}_u^{\mathcal{I}}|} \sum_{k \in \mathcal{N}_u^{\mathcal{I}}} \boldsymbol{\beta}_k \right)^2 + \sum_{i \in \mathcal{I}} \left(\boldsymbol{\beta}_i - \frac{1}{|\mathcal{N}_i^{\mathcal{U}}|} \sum_{k \in \mathcal{N}_i^{\mathcal{U}}} \boldsymbol{\alpha}_k \right)^2 \tag{5}$$

Through Equation 5, we aim to explicitly inject the neighborhood information into the users and items, leading to more robust representations of users/items and their neighbors.

*2) **Regularizer 2 – Distance Regularizer**:* The objective function shown in Equation 2 trains the user and item embeddings so that given a positive user–item interaction (u, i), the item embedding $\boldsymbol{\beta}_i$ is the nearest neighbor of the user embedding $\boldsymbol{\alpha}_u$ translated by the translation vector $\boldsymbol{r}_{ui}$; i.e., $\boldsymbol{\alpha}_u + \boldsymbol{r}_{ui} \approx \boldsymbol{\beta}_i$. In other words, the objective function (Equation 2) expects that the positive item i will be pulled toward user u by pushing the negative item j away from user u, which is in fact what is done in CML. However, the relations become more complex as the number of user–item interactions grows, and it is crucial to guarantee that the actual distance between them is small. Therefore, we introduce a second regularizer named $reg_{\text{dist}}(\Theta)$ to explicitly pull the item embedding closer to the translated user embedding as follows:

$$reg_{\text{dist}}(\Theta) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{N}_u^{\mathcal{I}}} -s(u,i) = \sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{N}_u^{\mathcal{I}}} \|\boldsymbol{\alpha}_u + \boldsymbol{r}_{ui} - \boldsymbol{\beta}_i\|_2^2 \tag{6}$$

Notably, $reg_{\text{dist}}(\Theta)$ is equivalent to the loss $\mathcal{L}_{\text{pull}}$, which is introduced in the paper that proposed CML [4]. However, CML does not employ $\mathcal{L}_{\text{pull}}$ because *"an item can be liked by many users and it is not feasible to pull it closer to all of them"* as the authors mentioned in Section 3.1 of their paper [4]. This infeasibility is essentially caused by the fact that each user is projected to a single point, and we argue that translating a user to multiple points by introducing the user–item specific translation vectors as in our method makes it feasible to pull the item closer to all of the translated users, allowing TransCF to employ $reg_{\text{dist}}(\Theta)$.

Final Objective Function. Given the margin–based pairwise ranking function (Equation 2) and the two regularizers (Equations 5 and 6), our final objective function $\mathcal{J}(\Theta)$ to minimize is as follows:

$$\mathcal{J}(\Theta) = (\mathcal{L}(\Theta) + \lambda_{\text{nbr}} \cdot reg_{\text{nbr}}(\Theta) + \lambda_{\text{dist}} \cdot reg_{\text{dist}}(\Theta))$$

where λ_{nbr} and λ_{dist} are regularization coefficients for the neighborhood regularizer and the distance regularizer, respectively. We compute the gradient for parameters in $\Theta = \{\boldsymbol{\alpha}_u, \boldsymbol{\beta}_i\}$, and update them by using mini-batch stochastic gradient descent (SGD) with learning rate η as follows: $\Theta \leftarrow \Theta - \eta \times \frac{\partial \mathcal{J}(\Theta)}{\partial \Theta}$. As our focus is to verify the benefit of translation mechanism for recommendation, we do not adopt advanced negative sampling techniques [18]–[21] for the model training. Instead, for each user $u \in \mathcal{U}$, we randomly generate 100 samples of $\{(i,j) | i \in \mathcal{N}_u^{\mathcal{I}} \cap j \notin \mathcal{N}_u^{\mathcal{I}}\}$ in every epoch. Moreover, as done in CML, we apply regularizations on $\boldsymbol{\alpha}_*$ and $\boldsymbol{\beta}_*$ after each epoch by bounding them within a unit sphere to mitigate 'curse of dimensionality' issue [8], [9]: $\|\boldsymbol{\alpha}_*\|^2 \leq 1$ and $\|\boldsymbol{\beta}_*\|^2 \leq 1$, which is achieved by $\boldsymbol{\alpha}_* \leftarrow \boldsymbol{\alpha}_* / \max(1, \|\boldsymbol{\alpha}_*\|^2)$ and $\boldsymbol{\beta}_* \leftarrow \boldsymbol{\beta}_* / \max(1, \|\boldsymbol{\beta}_*\|^2)$.

IV. Experiments

The experiments are designed to answer the following research questions (RQs):

- **RQ 1** How does TransCF perform compared with other state-of-the-art competitors?
- **RQ 2** Are the newly introduced regularizers tailored to TransCF beneficial for the performance of TransCF?
- **RQ 3** Do the user–item specific translation vectors indeed translate the users close to their corresponding items (as illustrated in Figure 1)?
- **RQ 4** What is encoded in the translation vectors?
  – *Intensity / Heterogeneity* of user–item relationships.

TABLE I: Data Statistics. (Rat. denotes the range of ratings, and #Cat. denotes the number of unique item categories.)

Dataset	#Users	#Items.	#Inter.	Density	Rat.	#Cat.
Delicious	1,050	1,196	7,698	0.61%	-	-
Tradesy	3,352	5,547	32,710	0.13%	-	-
Ciao	6,760	11,166	146,996	0.19%	1-5	28
Amazon	59,089	17,969	332,236	0.03%	1-5	45
Bookcr	19,571	39,702	605,178	0.08%	1-10	-
Pinterest	55,187	9,329	1,462,895	0.28%	-	-
Flixster	69,482	25,687	8,000,690	0.45%	0.5-5.0	-

Datasets. We evaluate our proposed method on *seven* real-world datasets: Delicious [22], Tradesy [23], Ciao [15], Amazon Cellphone and Accessories (Amazon C&A) [16], Bookcrossing [24], Pinterest [25], and Flixster [26]. Delicous, Tradesy and Pinterest datasets contain implicit feedback records, i.e., bookmark, purchased and pinning history. Ciao and Amazon C&A datasets contain rating information of users given to items, and also contain item category information, which will later be used in our experiments pertaining to verifying the heterogeneity of user–item relationships. Bookcrossing dataset contains both the rating (from 1 to 10) information and the implicit feedback (denoted as 0) from users on items. Flixster dataset contains the rating (from 0.5 to 5.0 by an interval of 0.5) information. For datasets with rating information, we regard each observed rating as an implicit feedback record as done in previous research [10], [16], [18], [23], [27]–[29]. We use several explicit feedback datasets and convert them into implicit feedback to be able to experimentally ascertain that the translation vectors indeed encode the intensity of user–item relationships; this is not possible with pure implicit feedback datasets as the interactions are not labeled. We include Bookcrossing and Flixster datasets with 10-level ratings in addition to the 5-level ratings (Ciao and Amazon C&A) to verify that TransCF can better infer knowledge regarding the intensity of user–item relationships when finer grained user preferences are given. We remove users and items having fewer than five ratings. The statistics of the preprocessed datasets used in our experiments are summarized in Table I.

Evaluation Protocol and Metrics. We employ the widely used leave-one-out evaluation protocol [16], [18], [23], [27]–[30], whereby the last interacted item for each user is held out for testing, and the rest are used for training. Since it is time consuming to rank all the items in $\mathcal{I}$, we sample 99 items for each user that the user had not interacted with, and compute the ranking scores for those items plus the user's test item (100 in total for each user) [28], [30]. For each user, we additionally hold out the last interacted item from the training data for validation set on which we tune the hyperparameters for all the methods. As we are focused on recommendations for implicit feedback, we employ two ranking metrics widely used for evaluating the performance of recommender systems [28], [29]: hit ratio (H@N) and normalized discounted cumulative gain (N@N). H@N measures whether the item is present in the top-N list, whereas N@N is a position-aware ranking metric that assigns higher scores to hits at upper ranks.

Methods Compared. As TransCF is a pair-wise *1) learning-to-rank* method based on *2) metric learning* that employs *3) neighborhood information*, we choose the following baselines.

1) Learning-to-rank baselines.
   - Point-wise methods.
     – **eALS** [18]: The state-of-the-art MF–based method for implicit feedback that non-uniformly weights the unobserved interactions based on the item popularity.
     – **NeuMF** [28]: A pointwise neural collaborative filtering framework for implicit feedback that combines MF and multi-layer perceptron (MLP). We report the best results from among MF, MLP, and NeuMF.
   - Pairwise methods.
     – **BPR** [27]: A pairwise learning-to-rank method for implicit feedback in which observed items are assumed to be highly preferred by users to unobserved items.
     – **AoBPR** [20]: An extension of BPR that samples popular items as negative feedback with a higher probability.
2) Neighborhood–based baselines.
   - **FISM** [10]: A neighborhood–based recommendation method based on MF in which a user is represented by the items that the user has interacted with as in Eqn. 4.
   - **CDAE** [11]: The state-of-the-art neighborhood–based method in which the user–item relationship is computed between a user and his neighbors, i.e., the items that the user has previously interacted with.
3) Metric learning–based baselines & Ablations of TransCF.
   - **CML** [4]: The state-of-the-art metric learning–based recommendation method for implicit feedback in which Euclidean distance is used for the scoring function. i.e., $s(u,i) = -\|\boldsymbol{\alpha}_u - \boldsymbol{\beta}_i\|_2^2$.
   - **TransCF**$^{\mathrm{dot}}$: An ablation of TransCF in which instead of Euclidean distance, inner product is used for the scoring function. i.e., $s(u,i) = (\boldsymbol{\alpha}_u + \boldsymbol{r}_{ui})^T \boldsymbol{\beta}_i$.
   - **TransCF**$^{\mathrm{alt}}$: Another ablation of TransCF in which the translation vector is computed by $\boldsymbol{r}_{ui} = f(\boldsymbol{\alpha}_u, \boldsymbol{\beta}_i)$. i.e., we employ the current user and item embeddings for constructing $\boldsymbol{r}_{ui}$ *instead of their neighborhoods*.

Implementation Details. For each data, the hyperparameters are tuned on the validation set by grid searches with $K \in \{8, 16, 32, 64, 128\}$, $\eta \in \{0.0005, 0.001, 0.005, 0.01, 0.05, 0.1\}$, $\gamma \in \{0.0, 0.1, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0\}$, $\lambda, \lambda_{\mathrm{nbr}}, \lambda_{\mathrm{dist}} \in \{0.0, 0.001, 0.01, 0.1\}$, where λ is the regularization coefficient for the baseline methods. We report the test performance measured using

TABLE II: Test performance of different methods. Best results are in bold face. (*Imp.* denotes the improvement of TransCF over the best competitor, which is CML.)

Datasets	Metrics	BPR	FISM	AoBPR	eALS	CDAE	NeuMF	CML	TransCF$^{\text{dot}}$	TransCF$^{\text{alt}}$	TransCF	*Imp.*
Delicious	H@10	0.1981	0.2203	0.2243	0.1992	0.1319	0.1164	0.2470	0.2150	0.2174	**0.2586**	4.70%
	H@20	0.3177	0.3391	0.3602	0.2942	0.2414	0.2171	0.3649	0.3377	0.3084	**0.3786**	3.75%
	N@10	0.1122	0.1124	0.1114	0.1035	0.0674	0.0558	0.1389	0.1101	0.1281	**0.1475**	6.19%
	N@20	0.1418	0.1424	0.1452	0.1271	0.0949	0.0789	0.1678	0.1412	0.1494	**0.1781**	6.14%
Tradesy	H@10	0.2481	0.2676	0.2597	0.2058	0.1652	0.1167	0.3031	0.2846	0.2648	**0.3198**	5.51%
	H@20	0.4174	0.4109	0.4256	0.3314	0.2867	0.2290	0.4413	0.4266	0.3823	**0.4505**	2.08%
	N@10	0.1248	0.1309	0.1300	0.1042	0.0831	0.0538	0.1685	0.1449	0.1466	**0.1767**	4.87%
	N@20	0.1673	0.1670	0.1715	0.1356	0.1136	0.0817	0.2031	0.1806	0.1760	**0.2095**	3.15%
Ciao	H@10	0.1569	0.2100	0.1873	0.1419	0.1770	0.1535	0.2085	0.2011	0.1991	**0.2292**	9.93%
	H@20	0.2811	0.3482	0.3146	0.2570	0.3153	0.2788	0.3337	0.3185	0.3270	**0.3740**	12.08%
	N@10	0.0751	0.1027	0.0891	0.0670	0.0862	0.0741	0.1053	0.1017	0.0989	**0.1167**	10.83%
	N@20	0.1063	0.1374	0.1209	0.0957	0.1208	0.1040	0.1358	0.1311	0.1309	**0.1525**	12.30%
Book-crossing	H@10	0.2425	0.2178	0.2563	0.1655	0.2244	0.2286	0.2885	0.2802	0.2828	**0.3329**	15.39%
	H@20	0.3761	0.3938	0.3916	0.2864	0.3610	0.3747	0.4053	0.3932	0.4069	**0.4744**	17.05%
	N@10	0.1250	0.1002	0.1338	0.0791	0.1164	0.1158	0.1663	0.1618	0.1578	**0.1865**	12.15%
	N@20	0.1585	0.1444	0.1676	0.1093	0.1506	0.1482	0.1956	0.1903	0.1890	**0.2221**	13.55%
Amazon C&A	H@10	0.2489	0.2470	0.2646	0.2161	0.2817	0.1317	0.3011	0.3003	0.3184	**0.3436**	14.11%
	H@20	0.3821	0.3782	0.3946	0.3480	0.4117	0.2390	0.4123	0.4184	0.4509	**0.4658**	12.98%
	N@10	0.1276	0.1247	0.1391	0.1064	0.1613	0.0613	0.1752	0.1648	0.1766	**0.2019**	15.24%
	N@20	0.1610	0.1577	0.1718	0.0739	0.1939	0.0880	0.2031	0.1945	0.2094	**0.2323**	14.38%
Pinterest	H@10	0.4759	0.4444	0.4921	0.3301	0.5244	0.4546	0.5378	0.5485	0.4899	**0.5504**	2.34%
	H@20	0.7564	0.6720	0.7618	0.5621	0.7393	0.6852	0.7771	0.7822	0.7514	**0.8108**	4.34%
	N@10	0.2034	0.2048	0.2143	0.1373	0.2644	0.2165	0.2558	0.2549	0.2259	**0.2580**	0.86%
	N@20	0.2744	0.2626	0.2827	0.1959	0.3188	0.2748	0.3166	0.3143	0.2922	**0.3242**	2.40%
Flixster	H@10	0.6836	0.5985	0.6904	0.6320	0.6797	0.6596	0.7009	0.6014	0.7123	**0.7309**	4.28%
	H@20	0.8087	0.7597	0.8124	0.7359	0.7973	0.7816	0.8081	0.7147	0.8163	**0.8374**	3.63%
	N@10	0.3701	0.2794	0.3830	0.3513	0.4526	0.4588	0.4608	0.3417	0.4704	**0.4986**	8.20%
	N@20	0.4020	0.3206	0.4140	0.3778	0.4824	0.4895	0.4881	0.3705	0.4969	**0.5257**	7.70%

the hyperparameter values that give the best HR@10 on the validation set. For reliability, we repeat our evaluations five times with different random seeds for the model initialization, and we report mean test errors. We fix the number of samples in a mini-batch to 1000 for mini-batch SGD.

A. Performance Analysis

*1) **Recommendation performance (RQ 1)**:* Table II shows the performance of different methods in terms of various ranking metrics. We have the following observations from Table II. 1) We observe that CML outperforms the MF–based competitors (BPR, FISM, AoBPR, eALS, and NeuMF). This is consistent with the previous work [4], indicating that metric learning approaches overcome the inherent limitation of MF by learning a metric space wherein the triangle inequality is satisfied. 2) We observe that TransCF considerably outperforms the state-of-the-art competitor, namely CML, by up to 17.05% (achieved for HR@20 on Bookcrossing dataset). This verifies the benefit of the translation vectors that translate each user toward items according to the user's relationships with those items. 3) TransCF$^{\text{alt}}$ generally performs worse than CML, which implies that the translation vectors should be carefully designed, or else the performance will rather deteriorate. 4) The superior performance of TransCF over TransCF$^{\text{alt}}$ confirms that incorporating the neighborhood information is indeed crucial in collaborative filtering [1], [10], [11]. 5) TransCF$^{\text{dot}}$ generally performs worse than CML, which verifies that the inner product operation limits the performance despite the benefit of the translation mechanism adopted to TransCF$^{\text{dot}}$. 6) By further comparing HR@10 of TransCF$^{\text{alt}}$ on Delicious and Bookcrossing datasets (0.2174 and 0.2828 in Table II) with HR@10 of TransCF without any regularization (0.2388 and 0.2983 in Table III when $\lambda_{\text{dist}} = \lambda_{\text{nbr}} = 0.0$), we observe that TransCF without the regularizers still outperforms TransCF$^{\text{alt}}$. This again verifies that the neighborhood information itself is indeed beneficial even without the help of the neighborhood regularizer.

TABLE III: Effect of the regularization coefficients.

Delicious (HR@10)		λ_{nbr} = 0.0	0.001	0.01	0.1
λ_{dist}	0.0	0.2388	0.2388	0.2401	0.2454
	0.001	0.2375	0.2401	0.2401	0.2454
	0.01	0.2401	0.2401	0.2427	0.2467
	0.1	0.2520	0.2520	0.2533	**0.2586**

Bookcrossing (HR@10)		λ_{nbr} = 0.0	0.001	0.01	0.1
λ_{dist}	0.0	0.2983	0.3033	0.3227	0.3185
	0.001	0.3038	0.3074	0.3236	0.3181
	0.01	0.3213	0.3219	0.3264	0.3187
	0.1	**0.3329**	0.3329	0.3324	0.3151

*2) **Benefit of regularizers (RQ 2)**:* Table III shows the effect of the regularization coefficients on the performance of TransCF on Delicious and Bookcrossing datasets, where λ_{nbr} and λ_{dist} denote the strengths of the neighborhood regularizer (reg_{nbr}) and of the distance regularizer (reg_{dist}), respectively. Larger values imply a stronger contribution of the regularizer to the model, and $\lambda_* = 0.0$ indicates no regularization. We have the following observations: 1) Both regularizers are indeed beneficial for the model performance, and their impact varies across different datasets. 2) Although reg_{nbr} is beneficial, its impact on the model performance decreases as the reg_{dist} dominates the model; i.e., as λ_{dist} increases. 3) reg_{dist} is more helpful for the model performance compared with reg_{nbr}; the benefit of reg_{dist} becomes more clear on Bookcrossing dataset in which the user–item relations

are more complex compared with Delicious dataset. This confirms that explicitly pulling each translated user toward all of the positive items rather than merely pushing negative items away from each user helps to model the complex user–item interactions, which aligns with our motivation for the distance regularizer reg_{dist} as mentioned in Section III-D2.

TABLE IV: Analysis on the user–item specific translation vectors. (*Obs./Unobs.* denotes results on observed/unobserved user–item interactions.)

Dataset	*Obs.*	*Unobs.*	Dataset	*Obs.*	*Unobs.*
Delicious	64.63%	43.75%	Amazon	75.57%	31.96%
Tradesy	56.02%	43.01%	Pinterest	36.25%	33.08%
Ciao	54.63%	38.42%	Flixster	22.24%	2.88%
Bookcr.	55.42%	35.57%			

B. Qualitative Evaluations

*1) **Benefit of translation vectors (RQ 3)**:* In this section, we conduct experiments to verify whether the translation vectors learned by TransCF indeed translate each user closer to the observed (positive) items as in **Toy example** illustrated in Figure 1. To this end, for each user u and his observed item i, we check whether the following holds:

$$\|\boldsymbol{\alpha}_u - \boldsymbol{\beta}_i\|_2^2 > \|\boldsymbol{\alpha}_u + \boldsymbol{r}_{ui} - \boldsymbol{\beta}_i\|_2^2 \tag{7}$$

That is, we expect the distance between the item embedding vector $\boldsymbol{\beta}_i$ of observed item i and the translated user embedding vector $(\boldsymbol{\alpha}_u + \boldsymbol{r}_{ui})$ to be smaller than the distance between the item embedding vector $\boldsymbol{\beta}_i$ and the user embedding vector $\boldsymbol{\alpha}_u$ before translation. We calculate the percentage of observed/unobserved user–item interaction pairs that satisfy Equation 7 among all possible observed/unobserved pairs. We sample as many unobserved items as the number of observed items for each user for the comparisons. Here, we expect more observed pairs to satisfy Equation 7 than unobserved pairs. Table IV shows the results on the seven datasets.

We observe that users are generally translated closer toward their observed (positive) items, and at the same time translated farther away from the unobserved (negative) items. For instance, for Amazon C&A dataset, 75.57% of the observed user–item interactions satisfy Equation 7, whereas only 31.96% of the unobserved interactions satisfy it, which conversely implies that users in 68% ($\approx 100 - 31.96$) of the unobserved interactions translate users farther away from the unobserved items. We also note that for Flixster dataset, although only 22.24% of the observed interactions satisfy Equation 7, the overwhelming majority of the unobserved interactions, i.e., 97% ($\approx 100 - 2.88$), violate it, which means that the effect of users being translated away from the unobserved items results in placing the translated users relatively closer toward their observed items. Hence, we argue that by simultaneously translating a user toward his relevant items and away from his irrelevant items, TransCF achieves superior recommendation performance.

*2) **What is encoded in the translation vectors? (RQ 4)**:* In this section, we investigate why TransCF outperforms the state-of-the-art methods. For this purpose, we conduct various experiments with the translation vectors to verify our claims that the translation vectors encode the i) **intensity**, and the ii) **heterogeneity** of user–item relationships in implicit feedback.

TABLE V: Rating classification accuracy using translation vectors. (*Rand* denotes a random classifier, and RF denotes the random forest classifier [31].)

Acc.(%)	Ciao		Amazon		BookCr.[3]		Flixster	
	Rand	RF	*Rand*	RF	*Rand*	RF	*Rand*	RF
CML	|	50.3	|	50.1	|	39.1	|	20.5
TransCF$^{\text{emb}}$	19.9	50.3	20.1	50.3	13.8	40.1	10.0	20.5
TransCF	|	**53.0**	|	**50.8**	|	**43.7**	|	**23.4**
vs. CML	-	5.3	-	1.5	-	11.7	-	14.2

***Intensity* of user–item interactions.** With regard to the first claim, i.e., intensity, we assume that the rating information is a proxy for the intensity of user–item relationships. In other words, the higher the rating of an item given by a user, the higher the intensity of the user–item relationship. Since Ciao, Amazon C&A, Bookcrossing and Flixster datasets contain rating information, we use them to study whether we can conversely infer some knowledge about ratings using the translation vectors *even though the ratings are not utilized during the model training*. To this end, we perform rating classification to predict the rating of user u on item i given the user–item specific translation vector $\boldsymbol{r}_{ui}$ as input; for instance, there are five classes for datasets with ratings ranging from 1 to 5. Note that we perform classification instead of regression to clearly emphasize the difference between the numbers; the RMSE values of regression usually differ in the second decimal place [1]. To ascertain the benefit of the translation vectors learned by TransCF, i.e., $\boldsymbol{r}_{ui}^{\text{TransCF}}$, we compare the rating classification performance with that of CML. Since CML does not model the translation vectors [4], we alternatively define the translation vectors for CML, i.e., $\boldsymbol{r}_{ui}^{\text{CML}} := (\boldsymbol{\alpha}_u - \boldsymbol{\beta}_i)$, where $\boldsymbol{\alpha}_u$ and $\boldsymbol{\beta}_i$ are both trained by CML. Likewise, we additionally define synthetic translation vectors $\boldsymbol{r}_{ui}^{\text{TransCF}^{\text{emb}}} := (\boldsymbol{\alpha}_u - \boldsymbol{\beta}_i)$ for TransCF , where $\boldsymbol{\alpha}_u$ and $\boldsymbol{\beta}_i$ are both trained by TransCF, and name this method TransCF$^{\text{emb}}$. Table V summarizes the classification results[4]. We observe that the rating classification accuracy on translation vectors of TransCF outperforms that of CML and TransCF$^{\text{emb}}$, and that the improvement is the greatest on Flixster dataset. This implies that the **intensity** of user–item relationships is encoded in the translation vectors learned by TransCF, which explains the superior performance of TransCF as shown in Table II. However, we note that the improvements of classification accuracies compared with CML are not sufficiently large on Ciao and Amazon C&A datasets, reaching only 5.3% and 1.5%, respectively. This motivates us to further investigate the translation vectors with regard to the intensity of user–item relationships.

To further study the translation vectors, we group the results of the observed user–item interactions from Table IV according to their ground truth ratings, and calculate the percentage

[3]We regard ratings of less than 4 as a single class as they account for only 3.86%.

[4]We balance the class distribution by randomly sampling a fixed number of samples (num. samples in the minority class) from each class. We then perform 5-fold cross validation. We use the default setting of the random forest classifier in Scikit-learn [32], but the accuracies could be further improved by tuning the classifier.

TABLE VI: Analysis on the user–item specific translation vectors regarding the ratings. (*Eqn 7.* denotes the percentage of interactions that satisfy Equation 7, and Ptn. denotes the portion of each class.)

	Rating						
Ciao	1	2	3	4	5		
Eqn 7	61.5%	51.4%	55.4%	52.2%	55.4%		
Ptn.	4.8%	5.1%	11.4%	29.0%	49.7%		
Amazon	1	2	3	4	5		
Eqn 7	76.7%	76.3%	75.7%	75.2%	75.4%		
Ptn.	7.0%	5.7%	10.7%	20.1%	56.5%		
BookCr.	1-4	5	6	7	8	9	10
Eqn 7	55.3%	52.7%	55.2%	56.1%	57.2%	58.4%	58.8%
Ptn.	3.8%	10.3%	7.9%	17.0%	24.5%	17.3%	19.2%
Flixster	0.5-2.5	3.0	3.5	4.0	4.5	5.0	
Eqn 7	19.6%	19.9%	19.9%	22.2%	25.7%	27.2%	
Ptn.	17.3%	17.0%	16.8%	19.6%	10.1%	19.2%	

of interactions that satisfy Equation 7 in each rating group. Table VI shows the results. Under the assumption that higher ratings imply a higher intensity of user–item relationships, we expect more observed interactions to satisfy Equation 7 in higher rating groups. However, we observe that the results on Ciao and Amazon C&A datasets do not agree with our expectation, which explains the relatively small improvements in Table V. We conjecture that it is inherently challenging to infer users' fine-grained preferences from the user–item interactions of Ciao and Amazon C&A datasets, because 1) the range of the ratings is relatively small (integers from 1 to 5), and more importantly, 2) the majority (over 75%) of interactions belong to ratings from 4 to 5. In contrast, for Bookcrossing[5] and Flixster datasets whose ratings are in a wider range (from 1 to 10, and from 0.5 to 5.0, respectively), and relatively evenly distributed throughout the rating classes, the percentage of interactions satisfying Equation 7 increases as the ratings increase from 5 to 10 (for Flisxter, from 3.0 to 5.0). These results on Bookcrossing and Flixster datasets (Table VI) are in line with the results in Table V, where the relative improvement of the classification results of TransCF is the highest for Bookcrossing and Flixster datasets. To summarize our findings from Tables V and VI, we conclude that the intensity of user–item relationships is indeed encoded in the translation vectors, and that it becomes clearer with datasets from which users' preferences can be more precisely inferred, e.g., Bookcrossing and Flisxter datasets.

However, up to this point, it is still uncertain why the improvements of TransCF on Ciao and Amazon C&A datasets are considerable in terms of the quantitative evaluation (Table II), even though the rating information (intensity) is not as clearly encoded in the translation vectors as in the case of Bookcrossing and Flixster dataset. Hence, in the following section, we study whether the translation vectors of TransCF trained on Ciao and Amazon C&A datasets encode meaningful information other than the rating information.

***Heterogeneity* of user–item interactions.** In this section, we investigate the translation vectors from a different perspective, i.e., heterogeneity. Recall that TransCF aims to translate each user toward multiple items according to the user's heterogeneous tastes in various item categories, implying that the translation vectors encode information related to item categories. To verify this, we study whether we can conversely infer the categories of items using the translation vectors, *even though the item category information is not utilized during the training of TransCF*. To this end, we label each translation vector $\boldsymbol{r}_{ui}$ with its corresponding category, and select vectors from the ten most frequently appearing categories to perform classification. Table VIIa shows the item category classification accuracy on the translation vectors[4]. We observe that TransCF considerably outperforms CML. This implies that the user–item specific translation vectors indeed encode the **heterogeneity** of the user–item relationships with regard to users' tastes in various item categories, which provides a justification for the superior performance of TransCF on Ciao and Amazon C&A datasets.

TABLE VII: Results of item category classification.

Dataset	Method	Rand.	Random Forest
Ciao	CML	10.01%	67.86±0.47%
	TransCF$^{\text{emb}}$		67.27±0.28%
	TransCF		**80.97**±0.73%
Amazon C&A	CML	10.40%	54.26±0.74%
	TransCF$^{\text{emb}}$		54.85±0.51%
	TransCF		**81.24**±0.46%

(a) Classification on translation vectors ($\boldsymbol{r}_{ui}$).

Dataset	Method	Rand.	Random Forest
Ciao	CML	10.92%	80.41±1.59%
	TransCF		81.61±1.54%
Amazon C&A	CML	9.40%	47.94±3.34%
	TransCF		47.90±2.54%

(b) Classification on item embeddings ($\boldsymbol{\beta_i}$).

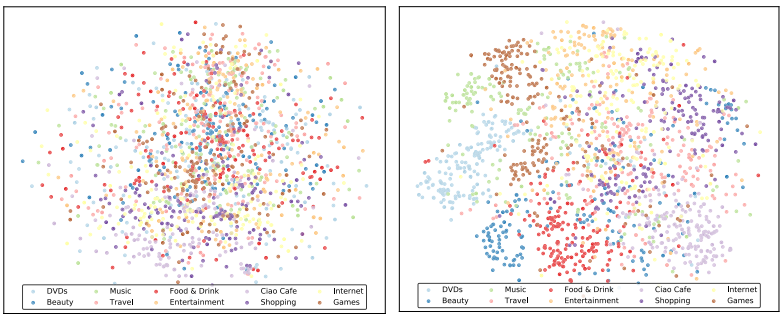
We further conduct another experiment on the item embedding vectors $\boldsymbol{\beta}_i$ to ascertain that the considerable performance improvement with TransCF is indeed derived from the translation vectors; not from the high-quality item embedding vectors. This time, we select items whose categories belong to the ten most frequent appearing categories. Table VIIb shows the classification accuracy on the item embedding vectors $\boldsymbol{\beta}_i$ for all $i \in \mathcal{I}$ trained by CML and TransCF. We observe that CML and TransCF show comparable classification performance, implying that the superior performance of TransCF is derived not from the high-quality item embedding vectors, but from the translation vectors.

To provide a more intuitive understanding of the numerical results shown in Table VIIa, we visualize in Figure 4 the translation vectors learned by CML and TransCF on Ciao dataset by using t-distributed Stochastic Neighbor Embedding[6](t-SNE) [33] with perplexity 30. Each point represents a translation vector, and the color represents the item category. We can clearly see that the translation vectors of TransCF are generally grouped together according to their corresponding categories, unlike CML; this supports the superior classification results of TransCF over CML in Table VIIa.

Lastly, based on the above observation that similar translation vectors encode similar information with regard to item cat-

[5]Note that for Bookcrossing dataset, the number of observed interactions whose ratings are less than 4 account for a small proportion (3.86%), and are therefore negligible.

[6]We sample 200 samples from each category for visualization.



(a) Visualization of $\boldsymbol{r}_{ui}^{CML}$

(b) Visualization of $\boldsymbol{r}_{ui}^{\text{TransCF}}$

Fig. 4: t-SNE visualization of translation vectors of Ciao dataset regarding the item categories.

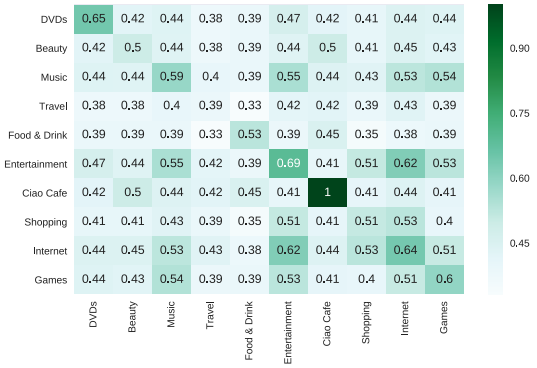


Fig. 5: Heat map of cosine similarity between translation vectors of Ciao dataset.

egories, we further investigate whether the translation vectors might even reveal the correlations among the item categories. To this end, we compute the sum of the cosine similarity scores between all pairs of translation vectors from each category. We use the same sampled translation vectors that were used for the visualization in Figure 4. Figure 5 shows the heat map of cosine similarity between translation vectors learned by TransCF on Ciao dataset with regard to the item categories. The number in each cell denotes the average cosine similarity score normalized by the overall largest score. We observe that the similarity score is generally the highest within the same item category (the diagonal line). Moreover, interestingly, semantically related categories also show high correlations. For example, "Entertainment" is highly correlated with "Internet", "Games", "Music", and "Shopping" all of which are related to entertainment. From the above analyses, we can conclude that TransCF can even determine the correlations among the item categories by encoding the heterogeneity of user–item relationships into the translation vectors, which again helps explain the superior performance of TransCF. This experiment also verifies our assumption illustrated in Figure 1 that the angles between the translation vectors reflect the heterogeneity of the user–item relationships regarding the user's tastes in various item categories.

V. Related Work

Recommendations for Implicit Feedback. Although explicit feedback such as rating or review comment is a valuable source of information that reveals user preferences, in most cases it is difficult to obtain a large quantity of such data. Hence, the vast majority of past research has focused on recommendations for implicit feedback [18], [19], [21], [23], [27], [34]–[36]. These methods generally adopt the matrix factorization (MF) technique, which uses the inner product to model the similarity of user–item pairs [1]. However, the inner product violates the triangle inequality, and thus it may fail to capture fine-grained user preferences [4].

Incorporating the neighborhood information [13] into CF has been shown to be effective for memory–based [14] and model–based [1], [10]–[12] CF. ItemCF [14] computes the similarity scores, such as Pearson Correlation and Cosine Similarity, between users based on the items they interacted with in the past. SLIM [12] and FISM [10] improve ItemCF by learning the item-item similarity directly from the data through factorizing the item similarity matrix as a product of two latent factor matrices. CDAE [11] is the state-of-the-art neighborhood–based CF method implemented by using denoising auto-encoder. However, previously proposed neighborhood–based CF methods are generally based on MF, and thus suffer from the violation of triangle inequality.

In the following, we briefly introduce metric learning approaches that address this limitation of MF.

Metric Learning. Metric learning [37] learns a distance metric that preserves the distance relation among the training data; i.e., it assigns shorter distances to semantically similar pairs. It has been popularized in various domains such as computer vision [38] and natural language processing [39]. Recently, metric learning approaches have been adopted in collaborative filtering to address the limitation of MF. Khoshneshin and Street firstly introduced the Euclidean embedding scheme for explicit feedback-based CF, a scheme in which users and items are embedded according to their Euclidean similarity rather than their inner product [5]. For music recommendation, Chen *et al*. represent songs as points in Euclidean space, and model the transition probability based on the Euclidean distance between songs [6]. For point-of-interest (POI) recommendation, Feng *et al*. model personalized check-in sequences by projecting each POI into one object in Euclidean space [7]. The above methods can be subsumed by the recently proposed CML [4], which is a general recommendation framework for implicit feedback. CML projects users and items into a common Euclidean space in which the similarity of a user–item pair is computed based on the Euclidean distance between the latent vectors of the user and of the item. Given user u and item i, the scoring function $s(u, i)$ of CML is computed by: $s(u, i) = -\|\boldsymbol{\alpha}_u - \boldsymbol{\beta}_i\|_2^2$. In spite of its state-of-the-art performance, CML projects each user to a single point, and thus it does not suffice for modeling the intensity and the heterogeneity of the user–item relationships in implicit feedback.

Knowledge Graph Embedding. Knowledge graph embedding refers to the projection of entities and relations in knowledge graphs into low-dimensional vector spaces. Among various knowledge graph embedding methods, translation–based methods [8], [9] have shown state-of-the-art perfor-

mance and scalability compared with traditional factorization–based embedding methods [40], [41]. Recently, the translation mechanism has been adopted for recommendation algorithms. Zhang *et al*. [42] integrate collaborative filtering and a knowledge base for recommendation by adopting TransR [9] to extract the structural knowledge of items from knowledge graphs. He *et al*. [29] embed items into a transition space, where each user is modeled by a translation vector to model the transition from the previously consumed item to the next item. However, our work is distinguished from the above methods in that we adopt the translation mechanism to model the latent relationships of implicit user–item interactions rather than to extract some knowledge from knowledge graphs, and we do not model the temporal information. Hence, we do not compare them with TransCF in our experiments.

VI. Conclusion

Each implicit user–item interaction encodes a different intensity of user–item relationship, and the relationship is heterogeneous regarding the user's taste in different item categories. In this paper, we propose a novel metric learning–based recommendation method called TransCF that captures not only the intensity and the heterogeneity of user–item relationships in implicit feedback, but also the complex nature of CF, all of which have been overlooked by previous metric learning–based recommendation approaches. TransCF employs the neighborhood information of users and items to construct translation vectors whereby a user is translated toward items according to his relationships with the items. Through extensive experiments, we demonstrate that TransCF considerably outperforms several state-of-the-art methods by generating meaningful translation vectors. Regarding possible directions for future studies, refer to Section III-C1.

Acknowledgment: NRF-2016R1E1A1A01942642, NRF-2017M3C4A7063570, IITP-2018-2011-1-00783, IITP-2018-0-00584, and SKT.

References

[1] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of SIGKDD*. ACM, 2008.

[2] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-based systems*, 2013.

[3] P. Ram and A. G. Gray, "Maximum inner-product search using cone trees," in *Proceedings of SIGKDD*. ACM, 2012.

[4] C.-K. Hsieh, L. Yang, Y. Cui, T.-Y. Lin, S. Belongie, and D. Estrin, "Collaborative metric learning," in *Proceedings of WWW*, 2017.

[5] M. Khoshneshin and W. N. Street, "Collaborative filtering via euclidean embedding," in *Proceedings of RecSys*. ACM, 2010.

[6] S. Chen, J. L. Moore, D. Turnbull, and T. Joachims, "Playlist prediction via metric embedding," in *Proceedings of SIGKDD*. ACM, 2012.

[7] S. Feng, X. Li, Y. Zeng, G. Cong, Y. M. Chee, and Q. Yuan, "Personalized ranking metric embedding for next new poi recommendation," in *Proceedings of AAAI*. AAAI Press, 2015.

[8] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in NIPS*, 2013.

[9] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proceedings of AAAI*. Citeseer, 2015.

[10] S. Kabbur, X. Ning, and G. Karypis, "Fism: factored item similarity models for top-n recommender systems," in *Proceedings of SIGKDD*. ACM, 2013.

[11] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-n recommender systems," in *Proceedings of WSDM*. ACM, 2016.

[12] X. Ning and G. Karypis, "Slim: Sparse linear methods for top-n recommender systems," in *Proceedings of ICDM*. IEEE, 2011.

[13] C. Desrosiers and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," in *Recommender systems handbook*. Springer, 2011.

[14] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of WWW*. ACM, 2001.

[15] J. Tang, H. Gao, and H. Liu, "mTrust: Discerning multi-faceted trust in a connected world," in *Proceedings of WSDM*. ACM, 2012.

[16] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *Proceedings of WWW*. International World Wide Web Conferences Steering Committee, 2016.

[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[18] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proceedings of SIGIR*. ACM, 2016.

[19] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," in *Proceedings of ICDM*. IEEE, 2008.

[20] S. Rendle and C. Freudenthaler, "Improving pairwise learning for item recommendation from implicit feedback," in *Proceedings of WSDM*. ACM, 2014.

[21] J. Weston, S. Bengio, and N. Usunier, "Wsabie: scaling up to large vocabulary image annotation," in *Proceedings of IJCAI*. AAAI Press, 2011.

[22] I. Cantador, P. Brusilovsky, and T. Kuflik, "2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011)," in *Proceedings of RecSys*, 2011.

[23] R. He and J. McAuley, "Vbpr: Visual bayesian personalized ranking from implicit feedback." in *AAAI*, 2016.

[24] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, "Improving recommendation lists through topic diversification," in *Proceedings of WWW*. ACM, 2005.

[25] X. Geng, H. Zhang, J. Bian, and T.-S. Chua, "Learning image and user features for recommendation in social networks," in *Proceedings of ICCV*, 2015.

[26] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proceedings of RecSys*. ACM, 2010.

[27] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of UAI*. AUAI Press, 2009.

[28] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th WWW*, 2017.

[29] R. He, W.-C. Kang, and J. McAuley, "Translation-based recommendation," in *Proceedings of RecSys*. ACM, 2017.

[30] H.-J. Xue, X.-Y. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," 2017.

[31] L. Breiman, "Random forests," *Machine learning*, 2001.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *JMLR*, 2011.

[33] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *JMLR*, 2008.

[34] W. Pan and L. Chen, "Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering," in *Proceedings of AAAI*, 2013.

[35] J. Lee, S. Bengio, S. Kim, G. Lebanon, and Y. Singer, "Local collaborative ranking," in *Proceedings of WWW*. ACM, 2014.

[36] C. C. Johnson, "Logistic matrix factorization for implicit feedback data," in *NIPS*, 2014.

[37] L. Yang, "Distance metric learning: A comprehensive survey," 2006.

[38] A. Bellet, A. Habrard, and M. Sebban, "A survey on metric learning for feature vectors and structured data," *arXiv preprint arXiv:1306.6709*, 2013.

[39] G. Lebanon, "Metric learning for text documents," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

[40] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proceedings of ICML*, 2011.

[41] R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski, "A latent factor model for highly multi-relational data," in *Advances in NIPS*, 2012.

[42] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of SIGKDD*. ACM, 2016.