AdaCML: Adaptive Collaborative Metric Learning for Recommendation

Tingting Zhang[1], Pengpeng Zhao[1(✉)], Yanchi Liu[2], Jiajie Xu[1], Junhua Fang[1], Lei Zhao[1], Victor S. Sheng[3], and Zhiming Cui[4]

[1] Institute of Artificial Intelligence, School of Computer Science and Technology, Soochow University, Suzhou, China
ttzhang7061@stu.suda.edu.cn,
{ppzhao,xujj,jhfang,zhaol}@suda.edu.cn
[2] Rutgers University, New Brunswick, USA
yanchi.liu@rutgers.edu
[3] University of Central Arkansas, Conway, USA
ssheng@uca.edu
[4] Suzhou University of Science and Technology, Suzhou, China
zmcui@mail.usts.edu.cn

Abstract. User preferences are dynamic and diverse in real world, while historical preference of a user may not be equally important as current preference when predicting future interests. As a result, learning the evolving user representation effectively becomes a critical problem in personalized recommendation. However, existing recommendation solutions often use a fixed user representation, which is not capable of modeling the complex interests of users. To this end, we propose a novel metric learning approach named Adaptive Collaborative Metric Learning (AdaCML) for recommendation. AdaCML employs a memory component and an attention mechanism to learn an *adaptive user representation*, which dynamically adapts to locally activated items. In this way, implicit relationships of user-item pairs can be better determined in the metric space and users' interests can be modeled more accurately. Comprehensive experimental results demonstrate the effectiveness of AdaCML on two datasets, and show that AdaCML outperforms competitive baselines in terms of Precision, Recall, and Normalized Discounted Cumulative Gain (NDCG).

Keywords: Recommender systems · Attention mechanism · Metric learning

1 Introduction

With the explosive growth of e-commerce and social media, we are living in an era of information explosion. Recommender Systems (RS) can alleviate the dilemma of information overload by delivering more relevant products to us. Collaborative Filtering (CF) is one of the most successful ways to build recommender systems and has received significant success in the past decades [1,11,17].

G. Li et al. (Eds.): DASFAA 2019, LNCS 11447, pp. 301–316, 2019.
https://doi.org/10.1007/978-3-030-18579-4_18

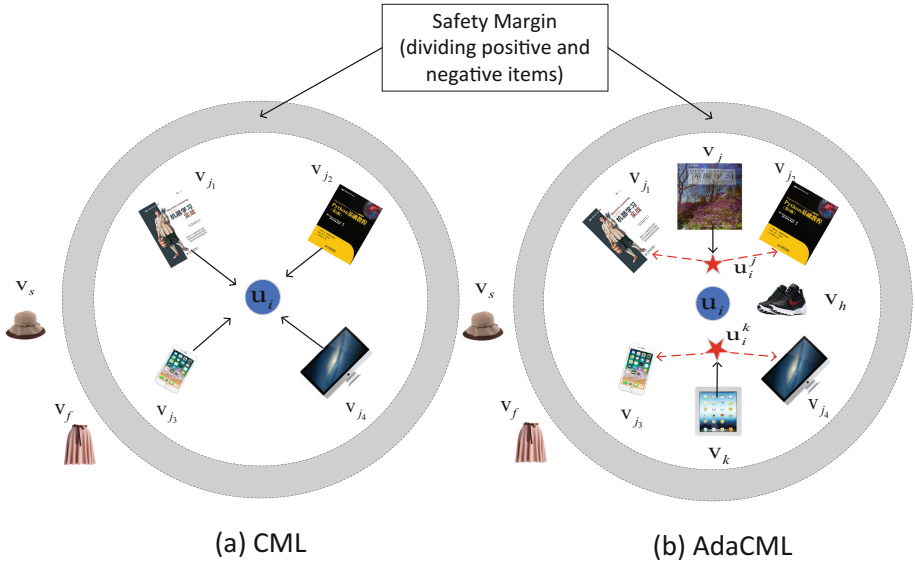

Fig. 1. The left image shows the standard collaborative metric learning method, and the right image denotes the adaptive user representation that can dynamically adapt to related items.

Specifically, it considers users' historical interactions and makes recommendations based on their similar preferences. Among all models of CF, Matrix Factorization (MF) [11] remains the most popular one, which tries to characterize both users and items in a low-dimensional latent vector space inferred from a historical user-item interaction matrix. However, MF captures users' preferences for items by performing the dot product operation that violates the triangular inequality, resulting in suboptimal performance [9]. The triangle inequality states that for any three objects, the sum of any two pairwise distances should be greater than or equal to the remaining pairwise distance [9]. This implies that if x is close to y and z, then y is also close to z. Therefore, the MF method based on the dot product operation can only capture the local preferences of the observed data (e.g., x is close to y and z), but fails to capture the potential preference information (e.g., y is also close to z). Hseih et al. [9] proposed Collaborative Metric Learning (CML) scheme that uses Euclidean distance to measure user-item relationships, which follows triangle inequality. CML not only captures user-item relationships, but also learns user-user similarity and item-item similarity in vector space. This collaborative metric learning (CML) method [9] has demonstrated highly competitive performance on many benchmark datasets.

Although CML obtains promising performance, it still faces some challenges. CML often projects users and items as fixed low-dimensional representation vectors into a user-item joint space. However, a fixed user representation greatly limits the ability of the CML method to model the diversities of users' interests.

This leads to a lack of flexibility in the CML model, which is unable to approach different candidate items discriminatively. In fact, users prefer different kinds of items, and only a subset of users' historical items is relevant to the candidate products. For instance, a user has previously purchased books, shoes, and badminton rackets, and then the user may buy badminton. His/Her purchase of badminton is more related to the badminton rackets than the books or the shoes previously purchased. Hence, it is difficult to use a fixed user representation to model the diversities of users' interests. As shown in Fig. 1, a user u_i liked a list of different kinds of items $\{v_{j_1}, v_{j_2}, v_{j_3}, v_{j_4}\}$. The user u_i and these items are represented as a latent vector in a user-item metric space. Since similar items are projected to similar locations, these items that user u_i liked form two clusters according to the types of items. Figure 1(a) shows that the essential idea of CML is to pull positive items (such as, items $\{v_{j_1}, v_{j_2}, v_{j_3}, v_{j_4}\}$) closer to the user u_i and push the negative items (i.e., items that the user did not like, such as, item v_s and v_f) away until they are beyond the safety margin. Given a list of candidate items $\{v_j, v_k, v_h\}$, the CML method with a fixed user representation will recommend the nearest unvisited items, such as item v_h, as shown in Fig. 1(b). Although item v_h is closer to user u_i than item v_j and item v_k, it is obvious that item v_j and item v_k are more reliable and valuable recommendations. Moreover, CML pursues performance while ignoring interpretability and significant insight, which is also a common problem of other methods [8,14].

To address the above problems, we propose an Adaptive Collaborative Metric Learning (AdaCML) method for recommendation. AdaCML uses a memory module and a novel attention mechanism to learn an *adaptive user representation* in a metric space. In this work, we assume that each user prefers different kinds of items and only a subset of the user's purchased products can reveal whether the user is interested in a candidate product. Hence, for different candidate items, an *adaptive user representation* adaptively selects locally activated items from user's historical items. As shown in Fig. 1(b), for the candidate item v_j, an *adaptive user representation* $\mathbf{u}_i^j$ is obtained by dynamically weighting the vectors of items v_{j_1} and v_{j_2}. Similarly, another *adaptive user representation* $\mathbf{u}_i^k$ is more related to items v_{j_3} and v_{j_4}. In this way, our model not only improves the performance but also enables interpretable recommendation results.

The main contributions of this paper are summarized as follows:

- The proposed AdaCML novelly applies an *adaptive user representation* to metric learning. The *adaptive user representation* can be dynamically adapted to locally activated items.
- We use a memory component and an attention mechanism to flexibly acquire *adaptive user representations* based on candidate items, which not only greatly improve the accuracy of the model, but also provide insight and interpretability for the model.
- We evaluate our AdaCML model on two real-world datasets. Our experimental results show that AdaCML outperforms various benchmarks with a significant margin.

The rest of the paper is organized as follows. We first define the problem of this paper in Sect. 2 and then present the framework of AdaCML in Sect. 3. After reporting our experimental study in Sect. 4, we introduce related work in Sect. 5. We finally conclude this paper in Sect. 6.

2 Problem Definition

In this section, we first introduce notations used in this paper, and then define the recommendation problem of our study. We apply bold capital letters (e.g., $\mathbf{X}$) and bold lowercase letters (e.g., $\mathbf{x}$) to represent matrices and vectors, respectively. Also, we employ non-bold letters (e.g., x, X) to denote scalars, and squiggle letters (e.g., $\mathcal{X}$) to denote sets. In this paper, we denote a set of users as $\mathcal{U} = \{u_1, u_2,..., u_N\}$ and an item set as $\mathcal{I} = \{v_1, v_2,..., v_M\}$, where N and M denote the number of users and items, respectively. For a user $u_i \in \mathcal{U}$, let $\mathcal{R}_i^+ = \{v_1, v_2, ..., v_{n_i}\}$ denotes the set of n_i items liked by user u_i and $\mathcal{R}_i^-$ denotes the remaining items. Then, given user set $\mathcal{U}$ and item set $\mathcal{I}$, for each user $u_i \in \mathcal{U}$, the recommendation task is defined as recommending a list of items from $\mathcal{R}_i^-$ that the user u_i may be interested in.

3 Adaptive Collaborative Metric Learning (AdaCML)

In this section, we first propose the overall architecture of AdaCML, and then introduce the details of each layer. Lastly, we will introduce the optimization details and time complexity analysis of AdaCML.

3.1 The Model Architecture

Figure 2 illustrates the architecture of the Adaptive Collaborative Metric Learning (AdaCML). The AdaCML model is based on a personalized memory module and a novel adaptive attention mechanism. Intuitively, users' interests are diverse, and users' historical records contain different types of items. The core issue of AdaCML is to design an *adaptive user representation* to characterize appetites of a user more accurately when the user interacts with different candidate items. Our model consists of five components, input layer, embedding layer, attention component, *adaptive user representation* and prediction layer. Specifically, the bottom input layer consists of two sparse vectors that denote user and item, respectively, where each element of each vector is either 0 or 1. Note that unlike the CML model that uses one-hot vector of each user as the input feature, AdaCML leverages multi-hot historical item IDs of each user as input features. Then the sparse vector is fed to the embedding layer, and each element with a value of 1 in the sparse vector corresponds to a dense vector. That is, feeding a user sparse vector to the embedding layer returns multiple dense vectors. Hence, we apply a memory matrix $\mathbf{M}$ to store the representation of each item in each user's historical records, so the memory matrix $\mathbf{M}$ naturally reflects the user's preferences. To design an *adaptive user representation*,

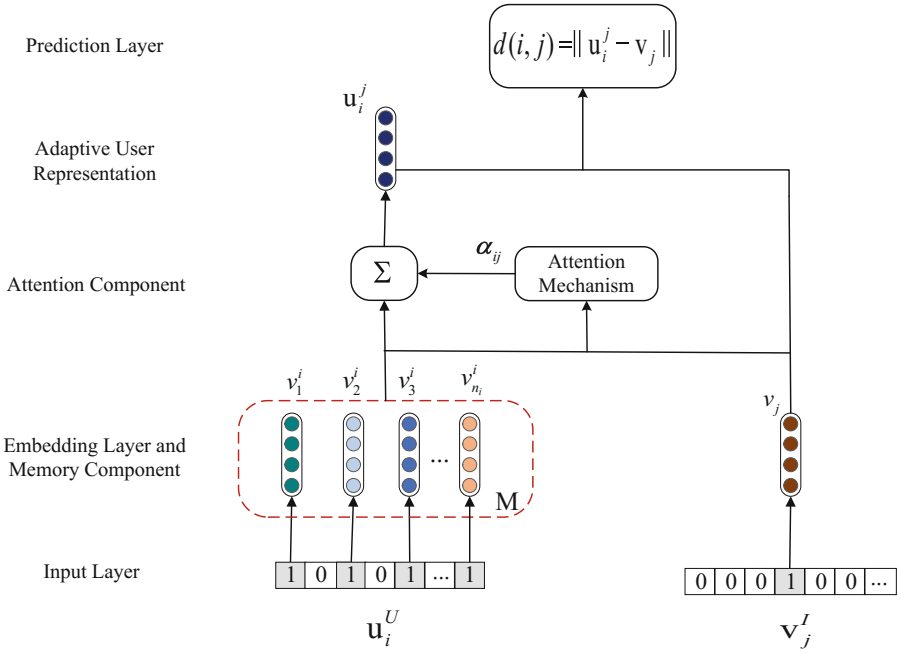

Fig. 2. The architecture of the adaptive collaborative metric learning.

we introduce an item-level attention mechanism on the memory matrix $\mathbf{M}$ to capture items in $\mathcal{R}_i^+$ relevant to item v_j. Here, we regard each item as a unit and model the impact of previously purchased items on the target item. After getting the *adaptive user representation*, we use the metric learning method to train and optimize our model AdaCML. Next, we will introduce each part of our model in details.

Input Layer. As shown in Fig. 2, the bottom input layer consists of two sparse vectors $\mathbf{u}_i^U$ and $\mathbf{v}_j^I$ that describe user u_i and item v_j, respectively. Moreover, the elements of the input vector of user u_i and item v_j are composed of 0 or 1. For $\mathbf{u}_i^U$, item IDs visited by user u_i are represented as 1, and the unvisited items are 0; for $\mathbf{v}_j^I$, where the index corresponding to item v_j is represented as 1 and the rest of positions are 0.

Embedding Layer and Memory Component. Above the input layer is the embedding layer that projects sparse representation to dense representation. The obtained each dense representation can be regarded as the latent feature vector for an item. Hence, for an item v_j, we formulate the item embedding vector as

$$\mathbf{v}_j = \mathbf{Q}^T \mathbf{v}_j^I, \tag{1}$$

where $\mathbf{Q} \in \mathbb{R}^{K \times M}$, denoting the latent vector matrix for all items. M is the number of items and K is the dimensionality of the user/item latent vector.

Since the user sparse vector $\mathbf{u}_i^U$ contains all item IDs previously accessed by the user u_i, this sparse vector input embedding layer will get multiple dense vectors. Therefore, we apply a memory module to store these dense feature vectors. This memory matrix $\mathbf{M}$ is constructed as

$$\mathbf{M} = \{\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_{n_i}\}, \tag{2}$$

where $\mathbf{M} \in \mathbb{R}^{K \times n_i}$. Each column of matrix $\mathbf{M}$ corresponds to a representation of an item in $\mathcal{R}_i^+$, so the matrix $\mathbf{M}$ reflects all interests of a specific user.

Attention Component. As mentioned in the introduction section, a fixed user vector representation may fail to convey diverse appetites of a user, resulting in the limitation of the representation ability of a model. Although the memory matrix $\mathbf{M}$ stores the information of all historical items purchased by each user, the goal of the attention mechanism is to selectively pick items that are closely related to the target item from $\mathbf{M}$ and then aggregate the representation of informative items to characterize the interests of a specific user [3]. Hence, given the user-item pair, (u_i, v_j), the AdaCML model uses an inner product to learn the similarity between the q-th item of the memory component and the target item v_j. Each element of the similarity score $\mathbf{w}_{ij}$ is defined as follows:

$$w_{ij}^q = \mathbf{m}_q^T \mathbf{v}_j, \forall q = 1, 2, ..., n_i, \tag{3}$$

where $\mathbf{m}_q \in \mathbb{R}^{K \times 1}$ is the q-th column vector of memory matrix $\mathbf{M}$. In order to normalize the similarity score $\mathbf{w}_{ij}$ to the attention weights distribution, we adopt the softmax function, which is a common practice in the attention network. We use α_{ij}^q to denote the q-th element of the attention vector $\boldsymbol{\alpha}$ as:

$$\alpha_{ij}^q = \frac{exp(w_{ij}^q)}{\sum_{h \in \mathcal{R}_i^+} exp(w_{ij}^h)}. \tag{4}$$

Note that the larger α_{ij}^q indicates the higher relevance between a target item v_j and a corresponding item v_q in $\mathbf{M}$.

Adaptive User Representation. In order to generate a dynamic and *adaptive user representation*, we use the attention vector $\boldsymbol{\alpha}$ to calculate a weighted sum representation of $\mathbf{M}$, i.e., adaptively selecting valuable pieces from the memory matrix $\mathbf{M}$. Hence, the *adaptive user representation* $\mathbf{u}_i^j$ is derived by weighting $\mathbf{M}$ according to the candidate item v_j as:

$$\mathbf{u}_i^j = \sum_{q \in \mathcal{R}_i^+} \mathbf{m}_q \alpha_{ij}^q, \tag{5}$$

where $\mathbf{m}_q$ corresponds to the q-th column of the memory matrix $\mathbf{M}$, and $\mathcal{R}_i^+$ denotes historical items liked by user u_i. It's worth noting that the proposed user representation is non-fixed and dependent on the candidate item v_j.

Prediction Layer. After computing an *adaptive user representation* $\mathbf{u}_i^j$ and an item vector $\mathbf{v}_j$, we learn these vectors according to their Euclidean distance.

Hence, when making predictions, we feed the final *adaptive user representation* $\mathbf{u}_i^j$ and the target item embedding $\mathbf{v}_j$ into the Euclidean distance:

$$d(i,j) = \|\mathbf{u}_i^j - \mathbf{v}_j\|, \tag{6}$$

where the *adaptive user representation* $\mathbf{u}_i^j$ is dependent on the attributes of the target item v_j and the user u_i's historical preferences. Here, the Euclidean function is used to measure the distance of user u_i and item v_j. It's more likely that the user u_i prefers item v_j if the distance between them is closer.

Overall, our proposed model AdaCML has the following advantages. First, considering the case where a given user's interests are diverse, we can leverage all items rated by the user to gain additional information. Second, with an attention mechanism, we allow each member of historical records of a user to contribute in *adaptive user representation* decision, where the contribution of each member is dependent on its feature and the property of a target item. Finally, an *adaptive user representation* can dynamically adapt to the user's interests based on the characteristics of the target item in an interpretable manner.

3.2 Model Inference

Finally, our model AdaCML adopts the hinge loss for optimization. For each positive user-item pair, (u_i, v_j), we randomly select several items from the user's unvisited records to form several negative user-item pairs. Similar to positive examples, negative pairs are represented as corresponding users and items hidden vectors in the same way. The hinge loss is defined as follow:

$$L = \sum_{(i,j)\in\mathcal{S}} \sum_{(i,j')\notin\mathcal{S}} \sigma_{ij}[\lambda + d(i,j)^2 - d(i,j')^2]_+, \tag{7}$$

where $\mathcal{S}$ is a set of positive user-item pairs, j is an item user u_i preferred, j' is an item he/she did not like, $[z]_+ = max(z, 0)$ denotes the standard hinge loss, and $\lambda > 0$ is the safety margin size that separates positive pairs and negative pairs. It is worth noting that because the user's representation depends on the attributes of the target item, positive user-item pairs do not share the same representation with negative user-item pairs. Namely, given a positive user-item pair (u_i, v_j) and a negative user-item pair $(u_i, v_{j'})$, the corresponding user feature vector are $\mathbf{u}_i^j$ and $\mathbf{u}_i^{j'}$, respectively.

Besides, our model, like CML, also uses a rank-based weighting scheme called Weighted Approximate-Rank Pairwise (WARP) loss, which was proposed by Weston et al. [16], to penalize positive items at a lower rank. As a result, we penalize the positive item v_j based on its rank by setting

$$\sigma_{ij} = log(rank_d(i,j) + 1), \tag{8}$$

where $rank_d(i,j)$ denotes the rank of item v_j in the recommended list of user u_i. At last, we employ regularization to constrain all the item embedding within a unit ball, i.e., $\|\mathbf{v}\|^2 \leq 1$, to prevent overfitting and ensure the robustness of the learned metric.

3.3 Time Complexity Analysis

In this subsection, we will analyze the time complexity of the AdaCML model. As we described in Sect. 3.1, the *adaptive user representation* component of AdaCML directly reflects the time cost of AdaCML in testing, i.e., Eq. (5). For a user-item pair, (u_i, v_j), the main time cost of an *adaptive user representation* $\mathbf{u}_i^j$ comes from the attention network. We use K to denote the embedding size, $|\mathcal{R}_i^+|$ to denote the number of historical interactions of user u_i, and a to denote the attention size. In this work, a is equal to K. And then we can obtain that the time complexity of each element of the similarity score $\mathbf{w}_{ij}$ is $O(aK)$. However, the similarity score $\mathbf{w}_{ij}$ includes $|\mathcal{R}_i^+|$ elements, so after using the softmax function to normalize $\mathbf{w}_{ij}$, the time complexity of the attention weight $\boldsymbol{\alpha}_{ij}$ becomes $O(aK|\mathcal{R}_i^+|)$. Finally, we use the weighted sum of the attention weight and the memory matrix to get the user adaptive representation. Hence, for each user, the time score of AdaCML is $O(aK|\mathcal{R}_i^+|^2)$. However, considering that the denominator term is shared across the computation of all items in $|\mathcal{R}_i^+|$, we only need to calculate it once and cache it for all items in $|\mathcal{R}_i^+|$. Therefore, the final time complexity of evaluating an AdaCML prediction is $O(aK|\mathcal{R}_i^+|)$.

4 Experiments

In this section, we will conduct experiments with the aim of answering the following research questions:

RQ1. How does our proposed model AdaCML perform, compared with state-of-the-art recommendation methods?
RQ2. How do the hyper-parameters influence the performance of AdaCML?
RQ3. Can the AdaCML model improve the interpretability of recommendation?

4.1 Experimental Settings

Datasets. We will study the performance of AdaCML on the Amazon dataset [7, 13]. Amazon[1] is an e-commerce platform and is widely used for product recommendation evaluation. We used two subsets of the Amazon review data as two datasets, namely Instant Video and Automotive. The Instant Video dataset contains users' ratings and review texts for the instant videos, while the Automotive dataset provides users' purchase records of various auto parts and accessories. Since we focus on the implicit feedback of users, we transformed the detailed ratings in the two datasets into 0 or 1, which indicates whether the user has rated the item. The statistics of the datasets are summarized in Table 1.

Baselines. We will compare our model AdaCML with following five state-of-the-art baseline models.

[1] http://jmcauley.ucsd.edu/data/amazon/.

Table 1. Datasets statistics

Dataset	Users	Items	Ratings	Sparsity
Instant video	1352	7785	16863	99.84%
Automotive	2928	1835	12962	99.76%

- **BPR** [14] is a strong baseline for building CF recommendation from implicit feedback data, which proposes a pair-wise optimization method to model the relative preference of users.
- **MLP** [8] applies a multi-layer perceptron above user and item embeddings to learn the scoring function. It is a standard neural network model with ReLu activates.
- **NeuMF** [8] is a unified framework fusing Matrix Factorization and Multi-Layer Perceptron. This model combines the linearity of Matrix Factorization and the non-linearity of Deep Neural Networks for modeling user-item interactions. Following the original work, we use the proposed tower structure.
- **MARS** [18] is a combination of Convolutional Neural Network (CNN) and attention mechanisms to learn an *adaptive user representation*. Since our experiment does not consider the text content of each item, we randomly initialize the item embedding based on Gaussian distribution to replace the operation of CNN extracting item features.
- **CML** [9] is a metric-based collaborative recommendation method, which solves the limitation of the inner product operate and obeys the triangle inequality.

Evaluation Metrics and Implementation Details. We divided each user's ratings into trainsets/testsets in a 70%/30% split. To evaluate these models described above, we repeated each individual model five times, and reported the average result of each model. In addition, we evaluated AdaCML and all baselines in terms of three different metrics, i.e., Precision, Recall, and Normalized Discounted Cumulative Gain (NDCG). We optimized all models with the Adam Optimizer. The learning rate of all models are tuned amongst {0.1, 0.01, 0.005, 0.001} and the batch size is fixed to 512. Without a special mention in this paper, we fixed the embedding size of all models to 32 for a fair comparison. Also, we set the margin size $\lambda = 1.5$ for AdaCML model, and $\lambda = 2$ for CML model on two datasets. We will investigate the influences of embedding size and margin size in Sect. 4.3. For AdaCML, the number of negative samples is fixed at 10, while the number of negative samples of all the baselines follows the setting in corresponding original papers. Both AdaCML and CML apply regularization by normalizing all user/item embedding to be constrained within the Euclidean ball. Also, for NeuMF and MLP, we employ a three-layer MLP that follows the original paper settings with the shape of (32, 16, 8). The MARS model can be understood as a variant of BPR, which takes the *user adaptive representation* into account, but it still uses dot product operations to capture the relationships between users and items.

Table 2. Experimental results of AdaCML and baselines. The best performance of each column (the larger the better) is in bold, and the second best is underlined.

Dataset	Method	@5			@10			@20		
		Recall	Precision	NDCG	Recall	Precision	NDCG	Recall	Precision	NDCG
Instant Video	BPR	0.01518	0.01102	0.03582	0.02595	0.00954	0.04635	0.04493	0.00825	0.05974
	MLP	0.01835	0.01295	0.03407	0.03161	0.01128	0.04726	0.04688	0.00871	0.05752
	NeuMF	0.02686	0.01975	0.05117	0.04347	0.01631	0.06503	0.07160	0.01333	0.08204
	CML	0.02573	0.01795	0.05069	0.04413	0.01614	0.06749	0.07720	0.01423	0.08804
	MARS	0.01870	0.01302	0.04478	0.02693	0.01002	0.05397	0.04228	0.00803	0.06495
	AdaCML	**0.03587**	**0.02626**	**0.07298**	**0.06553**	**0.02374**	**0.09645**	**0.10633**	**0.01923**	**0.11800**
Automotive	BPR	0.02023	0.01031	0.03240	0.03342	0.00856	0.04230	0.05298	0.00675	0.01198
	MLP	0.02412	0.01220	0.03873	0.03719	0.00927	0.04866	0.06770	0.00853	0.06476
	NeuMF	0.03930	0.01961	0.06037	0.06041	0.01508	0.07358	0.09445	0.01182	0.09003
	CML	0.03231	0.01600	0.04691	0.05456	0.01352	0.06131	0.08218	0.01032	0.07446
	MARS	0.04892	0.02377	0.07137	0.07771	0.01913	0.09181	0.11810	0.01470	0.10942
	AdaCML	**0.05584**	**0.02777**	**0.08087**	**0.09058**	**0.02249**	**0.10203**	**0.13654**	**0.01711**	**0.12161**

4.2 Performance Comparison (RQ1)

We compared the performance of AdaCML with five state-of-the-art baselines regarding Recall, Precision, and NDCG with cutoffs at 5, 10, and 20 on two datasets. Their overall recommendation performances are shown in Table 2. The experimental results reveal many interesting points as follows.

- Firstly, regardless of the datasets and the evaluation metrics, AdaCML always achieves the best performance, outperforming the state-of-the-art methods by a significant margin. This shows that by merging the *adaptive user representation* to metric learning, AdaCML can better model users' dynamic and diverse interests in a metric space, resulting in better recommendations.
- Secondly, pertaining to the performance of the baselines, we found that the performance of BPR is inferior on the two datasets. CML performs better than BPR. This indicates that modeling user-item relationships based on metric learning methods is more efficient than inner product. We can also see that NeuMF and CML perform similarly and outperform all other baselines on Instant Video. On the Automotive dataset, NeuMF also shows excellent performance and defeats most baselines except MARS. This shows that the NeuMF combines MF and MLP for a better performance. Moreover, we observed that MLP usually performs better than BPR, but falls short of CML on the two datasets in terms of Recall and Precision metrics. It is possible that NeuMF is a combination of multiple models to achieve well performance, and MLP alone may not yield good results. Besides, MARS performs better than BPR on the two datasets. This confirms that it is also effective to consider *adaptive user representation* in BPR.
- Finally, it is interesting to see that the performance of MARS is quite unstable. MARS performs quite well on the dataset Automotive while it does not perform well on Instant Video. This may be because the Instant Video is more relevant to real-time information, and only a small amount of items

previously accessed by the user are related to the current recommendation. However, AdaCML still achieves excellent results in this situation. This confirms the advantages of the metric learning method.

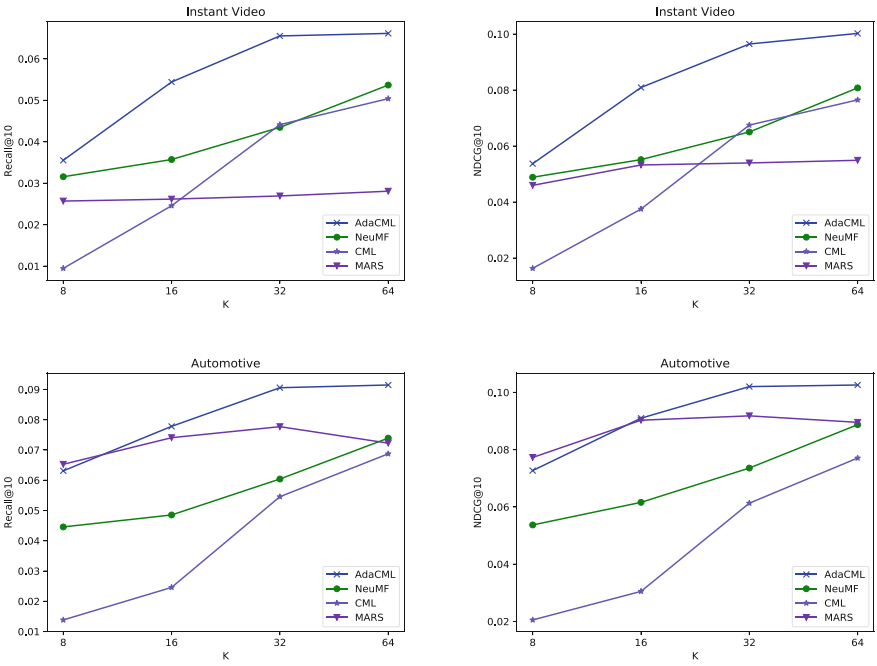

Fig. 3. Performance of these models under difference choices of the embedding size K.

4.3 Hyper-parameter Study (RQ2)

As we mentioned in Sect. 3, AdaCML has two crucial hyper-parameters (i.e., the embedding size K and the margin size λ). In this subsection, we will investigate the impact of the two hyper-parameters respectively. We only show the experimental results of the Recall@10 and NDCG@10 because of the space limitation.

Figure 3 shows the performance of theses methods at embedding size 8, 16, 32, and 64. Due to the poor performance of BPR and MLP, they are omitted in Fig. 3 to better highlight the performance differences among the rest of methods. On both datasets, AdaCML achieves the best performance under most cases, with the only exception of the embedding size setting as 8 on the dataset Automotive. This may be because the smaller embedding size is not enough to capture the complex relationships between users and items in the metric space. We also notice that CML performs extremely horribly with the embedding size setting as 8, which validates that a small hidden factor fails to represent users-items relationships. On the dataset Automotive, MARS reaches the peak when the embedding size is set as 32, followed by a degradation potentially due to

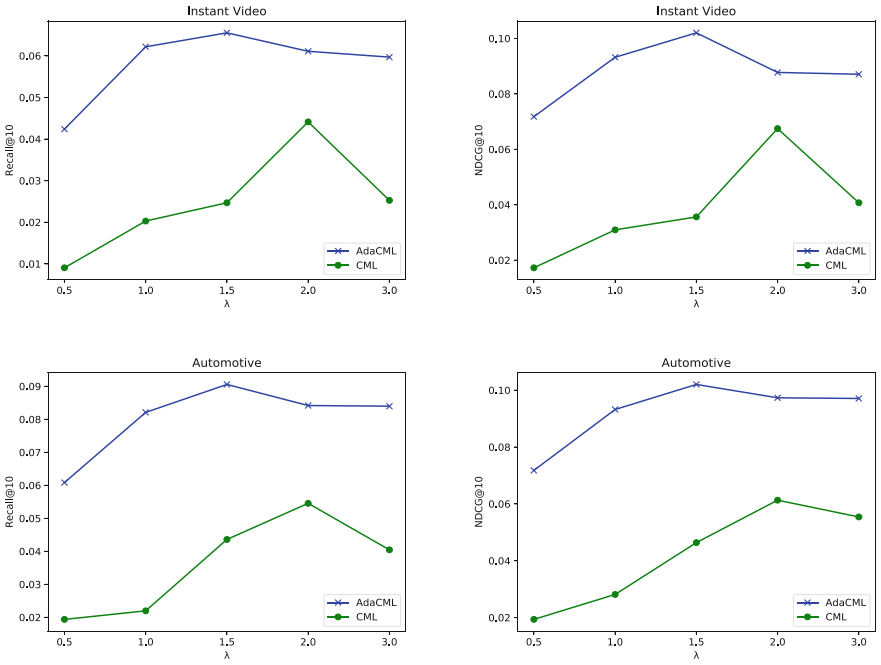

Fig. 4. Performance of AdaCML and CML under difference choices of the margin size λ.

overfitting. Compared with other models, the fluctuation of the experimental results of MARS is slight. This confirms that the method based on the dot product operation is not enough to capture the implicit relationships between users and items. Last but not least, as the embedding size increases, the performance of most models on the two datasets increases. This is because larger dimensions could capture more hidden factors of users and items, and then achieve better performance.

Figure 4 shows the performance of AdaCML and CML at margin size 0.5, 1.0, 1.5, 2.0, and 3.0. We can see that regardless of the setting of the margin size λ, AdaCML always outperforms CML. As we can see from Fig. 4, when $\lambda = 1.5$, AdaCML achieves the best performance, while CML reaches the peak at $\lambda = 2$. We attribute the phenomenon to the trade-off between positive items and negative items: too small λ can hardly distinguish between the positive and negative products, while too large λ brings much more noises (negative products) than useful products (positive products) within the safety margin.

4.4 The Interpretability of AdaCML (RQ3)

To gain a better insight into the interpretation ability of AdaCML, we conducted a qualitative experiment in this subsection. Table 3 shows the item-level attention weights concerning the items that users liked on the dataset Automotive. The

weights have been normalized to make a clear comparison. For example, for user #541, the target item #1486 is a positive example in the testset, while items #856, #85, and #237 are historical ones. We can see that AdaCML assigns higher weights on items #856 and #85, a lower weight on item #237, successfully evaluating the target item #1486 as the item desired by the user. However, for user #2048, the historical items #337 and #1014 have a higher weight, indicating that items #337 and #1014 are more relevant to the target item #397. To demonstrate the rationality, we further investigated the content of these items (i.e., the Automotive review text). Regarding user #541, we found that both the target item #1486 and the two higher attended items #856 and #85 are for better engine performances, while the lowest weight item #237 is about the windshield wiper. For user #2048, items #393 and #1014 with higher weights and the target item #397 are led lights, while the lowest weight item is Hoerr Racing Products. This is expected, because when predicting a user's preference on a target item, his/her historical items of the same category should have a larger impact than other less relevant items. Overall, this case study shows that AdaCML has great interpretability to explain its recommendation results.

5 Related Work

Our proposed AdaCML can be seen as an improvement of the popular Collaborative Metric Learning method (CML) [9]. It overcomes the drawbacks of CML by using a memory module and a novel attention mechanism to learn an *adaptive user representation* in a metric space. In this section, we will briefly discuss closely related work from two aspects, which are collaborative filtering and attention mechanism.

Table 3. A case study on the interpretability of AdaCML. The first column shows the user ID, the second column displays the target item ID, and the third column has three sub-columns, each of which contains a historical item and an attention weight, where the attention value is shown in parentheses.

User ID	Target item ID	Historical items		
		1	2	3
#541	#1486	#856 (0.398)	#85 (0.399)	#237 (0.203)
#2048	#397	#337 (0.393)	#1267 (0.239)	#1014 (0.368)

5.1 Collaborative Filtering

Collaborative Filtering (CF) is one of the most successful technique to build recommender systems, operating on explicit and implicit feedback data. Implicit

collaborative filtering is a method of learning user-item interactions from implicit data (e.g., buy or not buy, like or not), and also known as the one-class recommendation [7]. In implicit feedback data, user-item relationships are binary in nature (i.e., 1 if observed, and 0 otherwise). Bayesian Personalized Ranking (BPR) [14] is a well-known framework for disposing of the implicitness in CF. Instead of point-wise learning as in SVD [6], BPR uses a pair-wise learning algorithm to model the relative preferences of users, which makes the items that each user preferred more forward than those items he/she did not like. CML [9] is a recently proposed algorithm for CF and has demonstrated highly competitive performance on several benchmarks. CML learns a joint metric space that not only encodes users' preference for items but also learns user-user similarity and item-item similarity. Unlike most methods, CML uses Euclidean distance to measure the preferred relationships between users and items. CML minimizes the distance between each positive user-item interaction in Euclidean space. Hence, in this space, the items that users liked will keep coming close to users, and the uninterested items will be pushed further away. But CML still faces some challenges. Firstly, the representation of users is fixed and is insufficient to capture the complex and diverse interests. Secondly, CML lacks interpretability. To address these challenges, we propose AdaCML, which uses a memory module and a novel attention mechanism to learn an *adaptive user representation* in a metric space. It not only greatly improves the recommendation performance, but also provides the greater insight and the interpretability of recommendation. The details of AdaCML was explained in Sect. 3.

5.2 Attention Mechanism

Attention mechanism are prevalent in many domains, such as image/video caption [4,5], machine translation [12], and so on. The attention mechanism originates from the neural science studies by empirically demonstrating that human usually focus on specific parts of the target entity rather than using all available information [10]. The key idea of attention is to learn or select the attentive weights from a set of inputs. Higher (lower) weights indicate that the relevant components (members) are more informative (less informative) to generate the end output. Recently, the attention mechanism has been widely wielded to more accurately represent features in recommender systems [2,3,15]. For example, Chen et al. [2] proposed an interesting attention-driven factor model for recommendation. It adopts the attention model to measure the relevance of each part of each item. To get the representation for a multimedia object (e.g., image or video), Chen et al. [3] aggregated its components (e.g., regions or frames) with an attention network, where a similar attention mechanism is applied to aggregate interacted items to get a user representation for making recommendation.

Moreover, in most models, the attention mechanism is usually applied to calculate the importance of each feature in the item content to the user, which does not consider candidate items. This method generally captures the user's general preference and lacks flexibility, causing weak performance and poor explanation. Zheng et al. [18] utilized CNN and the attention mechanism to adaptively

capture local crucial historical items based on the contents of candidate items. However, MARS [18] still uses inner product to portray user-item interactions, which violates the triangular inequality. Hence, we present an AdaCML model that follows the triangle inequality. Our experimental results show that it has a better performance in capturing the diverse interests of users by combining the metric learning method with the attention mechanism.

6 Conclusion

In this paper, we propose an Adaptive Collaborative Metric Learning (AdaCML) model to perform a personalized recommendation for users. As we know, users' interests are diverse and involved in various domains. Most existing methods often focus on learning a fixed and static user representation. This is not enough to model the diverse interests of users. To tackle this problem, we used a memory component and an attention mechanism to learn an *adaptive user representation*, and utilized a metric learning algorithm to capture the relationships of user-item, user-user, and item-item. To the best of our knowledge, AdaCML is the first model to apply an *adaptive user representation* to a metric learning algorithm. We conducted extensive experiments on two real-world datasets and demonstrated that AdaCML could consistently outperform the state-of-the-art models by a significant margin.

Acknowledgments. This research was partially supported by the NSFC (61876117, 61876217, 61872258, 61728205, 61772356), the Suzhou Science and Technology Development Program (SYG201803) and the Blockshine Technology Corp.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. **17**(6), 734–749 (2005)
2. Chen, J., Zhuang, F., Hong, X., Ao, X., Xie, X., He, Q.: Attention-driven factor model for explainable personalized recommendation. In: The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 909–912. ACM (2018)
3. Chen, J., Zhang, H., He, X., Nie, L., Liu, W., Chua, T.S.: Attentive collaborative filtering: multimedia recommendation with item-and component-level attention. In: Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, pp. 335–344. ACM (2017)
4. Chen, L., et al.: SCA-CNN: Spatial and channel-wise attention in convolutional networks for image captioning. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6298–6306. IEEE (2017)
5. Cho, K., Courville, A., Bengio, Y.: Describing multimedia content using attention-based encoder-decoder networks. IEEE Trans. Multimedia **17**(11), 1875–1886 (2015)
6. Golub, G.H., Reinsch, C.: Singular value decomposition and least squares solutions. Numer. Math. **14**(5), 403–420 (1970)

7. He, R., McAuley, J.: 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 2016, pp. 507–517 (2016)
8. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web, WWW 2017, pp. 173–182 (2017)
9. Hsieh, C.K., Yang, L., Cui, Y., Lin, T.Y., Belongie, S., Estrin, D.: Collaborative metric learning. In: Proceedings of the 26th International Conference on World Wide Web, WWW 2017, pp. 193–201 (2017)
10. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Trans. Pattern Anal. Mach. Intell. **20**(11), 1254–1259 (1998)
11. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
12. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation (2015). arXiv preprint: arXiv:1508.04025
13. McAuley, J., Targett, C., Shi, Q., Van Den Hengel, A.: Image-based recommendations on styles and substitutes. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 43–52. ACM (2015)
14. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, pp. 452–461. AUAI Press (2009)
15. Vinh, T.D.Q., Pham, T.A.N., Cong, G., Li, X.L.: Attention-based group recommendation (2018). arXiv preprint: arXiv:1804.04327
16. Weston, J., Bengio, S., Usunier, N.: Large scale image annotation: learning to rank with joint word-image embeddings. Mach. Learn. **81**(1), 21–35 (2010)
17. Zhang, Y., Wang, H., Lian, D., Tsang, I.W., Yin, H., Yang, G.: Discrete ranking-based matrix factorization with self-paced learning. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 2758–2767. ACM (2018)
18. Zheng, L., et al.: Mars: memory attention-aware recommender system (2018). arXiv preprint: arXiv:1805.07037