Hindawi
Scientific Programming
Volume 2021, Article ID 8880331, 10 pages
https://doi.org/10.1155/2021/8880331

Research Article

Neighborhood Attentional Memory Networks for Recommendation Systems

Tianlong Gu,[1,2] **Hongliang Chen**,[1,3] **Chenzhong Bin**,[1,3] **Liang Chang**,[1,3] **and Wei Chen**[1,3]

[1]*School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin, China*
[2]*College of Information Science and Technology, College of Cyber Security, Jinan University, Guangzhou, China*
[3]*Guangxi Key Lab of Trusted Software, Guilin University of Electronic Technology, Guilin, China*

Correspondence should be addressed to Chenzhong Bin; binchenzhong@guet.edu.cn

Received 25 July 2020; Revised 19 January 2021; Accepted 4 February 2021; Published 17 February 2021

Academic Editor: Qianchuan Zhao

Deep learning systems have been phenomenally successful in the fields of computer vision, speech recognition, and natural language processing. Recently, researchers have adopted deep learning techniques to tackle collaborative filtering with implicit feedback. However, the existing methods generally profile both users and items directly, while neglecting the similarities between users' and items' neighborhoods. To this end, we propose the neighborhood attentional memory networks (NAMN), a deep learning recommendation model applying two dedicated memory networks to capture users' neighborhood relations and items' neighborhood relations respectively. Specifically, we first design the user neighborhood component and the item neighborhood component based on memory networks and attention mechanisms. Then, by the associative addressing scheme with the user and item memories in the neighborhood components, we capture the complex user-item neighborhood relations. Stacking multiple memory modules together yields deeper architectures exploring higher-order complex user-item neighborhood relations. Finally, the output module jointly exploits the user and item neighborhood information with the user and item memories to obtain the ranking score. Extensive experiments on three real-world datasets demonstrate significant improvements of the proposed NAMN method over the state-of-the-art methods.

1. Introduction

Information overload has become a challenge in the Internet era as the rapid increase in the user of information resources overloads many users' attentiveness. To alleviate the problem of information overload, recommendation systems have been widely adopted in many online services, such as e-commerce, social media sites, and online news. Collaborative filtering (CF) is one of the most popular and effective recommendation techniques. It establishes the relevance between users and items and relies on historical interactions (e.g., clicking and scoring) by assuming similar users will consume similar items.

Generally, there are three types of CF models: the latent factor models, the neighborhood-based approaches, and the hybrid models. The latent factor models, such as matrix factorization [1], represent a user or an item with a vector of latent features by projecting each user and item into a common low dimensional latent vector space. Typically, a user's interaction with an item is modeled as the inner product of the user latent vector and the item latent vector. Neighborhood-based methods form recommendation systems by identifying neighborhoods of similar users or items based on the previous interaction. In the early neighborhood-based methods, Amazon has achieved significant performance improvements by using collaborative filtering between items [2]. Latent factor models capture the global structure information of users and items but typically neglect the presence of strong associations between a few closely related users or items. In contrast, neighborhood-based methods capture the local structure information of users and items but often neglect the mass majority of ratings information outside the neighborhood. The above problems between these two classes of CF models lead to the development of hybrid models, such as SVD++ [3] and FM

[4], which integrate both neighborhood-based methods and latent factor models. Although these traditional hybrid models improve the accuracy of the recommendations, they primarily model the interaction between a user and an item in a linear way, such as with an inner product, which barely captures the higher-order complex user-item relations.

In recent years, increasing numbers of researchers have adopted the methods of deep learning [5–11] to study recommendation algorithms. The success of the recommendation algorithms based on deep learning has demonstrated the remarkable advantages of complex nonlinear transformations over traditional linear models. However, the existing deep learning methods such as DeepFM [12] and NeuMF [13] generally only consider the direct interaction between the target user and an item, with the result that the amount of feedback for a given user-item pair is sparse. Hence, we leverage all users who have rated a target item and all items that the target user has rated to gain additional insight on the existing user-item relations.

In this paper, we propose the neighborhood attentional memory networks (NAMN) incorporating two memory networks to realize the user neighborhood component and the item neighborhood component, which are called the user neighborhood memory network and the item neighborhood memory network, respectively. The memory components permit the encoding of rich user preference information and item attribute information. In the user neighborhood memory network, the attention mechanism assigns higher weights to specific subsets of users in the user neighborhood which share similar preferences with the target user. Then, the user neighborhood memory network utilizes these weights together with the corresponding user vectors in the user external memory form the vector representation of the user neighborhood. Analogously, we can obtain the vector representation of an item neighborhood by using an attention mechanism in the item neighborhood memory network. Further, to enhance the performance of recommendations, NAMN stacks multiple user and item neighborhood components to reason and infer more precise neighborhood information. Finally, NAMN use two nonlinear interactions, one between the user neighborhood information and the user memories and the other between the item neighborhood information and the item memories, to obtain the ranking score.

In summary, our main research contributions in this paper are as follows:

(i) We reveal that the user and item neighborhood information is crucial for improving recommendation performance.

(ii) We propose NAMN, which is motivated by recent progress in memory networks to deal with collaborative filtering based on implicit feedback. NAMN utilizes user and item neighborhood memory networks to capture users' and items' neighborhood relations and combine the user and item neighborhood information with the user and item memories in two nonlinear interactions to obtain a recommendation.

(iii) We conduct comprehensive experiments on three real-world datasets and validate the superiority of NAMN over seven state-of-the-art baselines and the effectiveness of both user neighborhood memory networks and item neighborhood memory networks.

2. Related Works

2.1. Deep Learning in Recommendation. In recent years, deep learning has been revolutionizing recommendation systems and has achieved excellent performance in many recommendation scenarios. Generally, recommendation systems based on deep learning can be classified into two categories: deep neural networks, used to process the raw features of users or items, or deep neural networks, used to model the interaction among users and items [7].

In one of the early works in this area, He et al. [13] proposed a neural collaborative filtering framework to address collaborative filtering based on implicit feedback by jointly learning a matrix factorization and a feedback neural network. Later, He et al. proposed a neural factorization machine in [14], which enhanced the implicit factorization machine by modeling higher-order and nonlinear interaction features. Xin et al. [15] proposed a convolutional factorization machine model, which seamlessly combined the automatic feature interaction modeling of factorization machines and the strong learning capabilities of 3D convolutional neural networks (CNN) and was able to capture high-order and nonlinear interaction signals. A multirelational memory network [16] is a unified neural learning framework that not only models fine-grained user-item relations but can also distinguish between feedback types according to the strength and diversity of the users' preferences.

Recently, there has been a surge of interest in applying attention mechanisms to recommendation tasks. Social attentional memory network [17] is a novel model for user-aware recommendations, which unifies the strengths of memory networks and attention mechanisms for modeling users' preferences by designing an attention-based memory module and a friend-level attention component. Yu et al. [18] improved the traditional structure of recurrent neural networks (RNNs) and proposed a time-aware controller and a content-aware controller, which can adaptively model users' long-term and short-term preferences. Gong and Zhang [19] approached hashtag recommendations with a CNN, augmented with an attention channel, to capture the most informative words. Most of the existing models based on neural attention rely on auxiliary information or context information, while our aim is to explore collaborative filtering with implicit feedback.

2.2. Memory Networks. Memory networks have made massive strides in many research fields, such as natural language processing, question answering, and knowledge tracking. Memory networks generally consist of two components: an external memory that stores long-term historical

information and a controller that performs read or write operations on the memory. The memory component can use memory matrices to store historical information, tracks long-term dependencies on historical data, and performs reasoning operations. The controller component manipulates these memories through a content-based or location-based addressing mechanism.

Zaremba and Sutskever [20] proposed a traditional deep learning model (e.g., RNN, LSTM, and GRU) encoding memory by hidden states or weights that were typically too small, and they were not compartmentalized enough to accurately remember facts from the past dense vectors. Weston et al. [21] proposed an initial framework of memory networks and demonstrated promising results by adding a series of memory units into the model, and it was applied to context-based question answering tasks. Sukhbaatar et al. [22] proposed an end-to-end memory network model that required less supervised data for training so that the model had good flexibility in handling various tasks for different language models. Huang et al. [23] applied the end-to-end memory network model to mention (@) recommendations in Twitter, by combining user interests with external memory and making personalized recommendations by combining tweet content, user history, and candidate user interests. Chen et al. [24] also used a user memory network for sequence recommendations. Considering that users' historical purchase behaviors may not be equally important when predicting users' future purchase interests, the external memory matrix in the network was used to enhance the memory to explicitly store and update users' historical records, which enhanced the expressiveness of the model. Lastly, it is worth mentioning that although the memory networks for collaborative filtering have been considered in a very recent method named collaborative memory network (CMN) [25], it only exploited a memory network to capture users' neighborhoods relations. Specifically, the output module of CMN integrates a nonlinear interaction between the user neighborhood information and the user and item memories to produce the ranking score. Distinct from CMN, we apply two memory networks to capture users' neighborhood relations and items' neighborhood relations, which jointly exploit the nonlinear interaction between the user neighborhood information and the user memories and the nonlinear interaction between the item neighborhood information and the item memories to obtain the ranking score. Our work is innovative in that it utilizes memory networks to capture user neighborhood relations and item neighborhood relations, simultaneously, in a collaborative filtering scenario.

3. Neighborhood Attentional Memory Networks

In this section, we introduce our model NAMN, whose framework is illustrated in Figure 1, and we present the related notations of NAMN in Table 1. NAMN consists of user neighborhood memory networks and item neighborhood memory networks that contain four memory states: user memory matrix, item memory matrix, user external memory matrix, and item external memory matrix. The model joins user and item neighborhood information with user and item memories in a nonlinear interaction way. The associative addressing scheme acts as a nearest neighbor similarity function, which allows it to learn semantically similar users who have accessed the current item and semantically similar items that have been accessed by the current user. The attention mechanisms permit learning an adaptive nonlinear weighting function in the user neighborhood and item neighborhood, respectively. Consequently, the users who are most similar in the user neighborhood and the items that are most similar in the item neighborhood contribute higher weights in the output module.

3.1. User Embedding and Item Embedding. The memory component consists of a user memory matrix $M \in R^{P\times d}$ and an item memory matrix $E \in R^{Q\times d}$, where P and Q denote the number of users and items respectively and d represents the dimensionality of each memory cell [25, 26]. Each user u is associated with a memory slot $m_u \in M$ that stores the user's specific preferences [25, 26]. Similarly, each item i is embedded in another memory slot $e_i \in E$ that encodes an item's specific attributes. We obtain a user preference vector q^1_{ui} where each dimension q^1_{uiv} represents the similarity of the target user u with the user v in the user neighborhood given item i and also forms an item attribute vector q^1_{iu} where each dimension q^1_{iuj} represents the similarity of the target item i with item j in the item neighborhood given user u as

$$q^1_{uiv} = m_u^T m_v + e_i^T m_v, \quad \forall v \in N(i), \tag{1}$$

$$q^1_{iuj} = e_i^T e_j + m_u^T e_j, \quad \forall j \in S(u), \tag{2}$$

where $N(i)$ is the user neighborhood set of all users who have accessed item i; $S(u)$ is the item neighborhood set of all items that have been accessed by user u. In formula (1), the first term represents the compatibility between the target user u and the user v, who is in the user neighborhood given item i. The second term computes the level of confidence that user v supports the recommendation of item i. In formula (2), the first term represents the compatibility between the target item i and the item j, which is in the item neighborhood given user u. The second term computes the level of confidence that item j is in line with the current user's preferences. Therefore, the associative addressing scheme identifies the internal memories with the highest similarity to the target user u in the user neighborhood given the specific item, while for a specific user, the associative addressing scheme identifies the internal memories with the highest similarity to the target item i in the item neighborhood.

3.2. User and Item Neighborhood Memory Networks. The attention mechanism learns two adaptive weighting functions to focus on a subset of influential users within the user neighborhood and on a subset of influential items within the item neighborhood to obtain the ranking score. Traditional

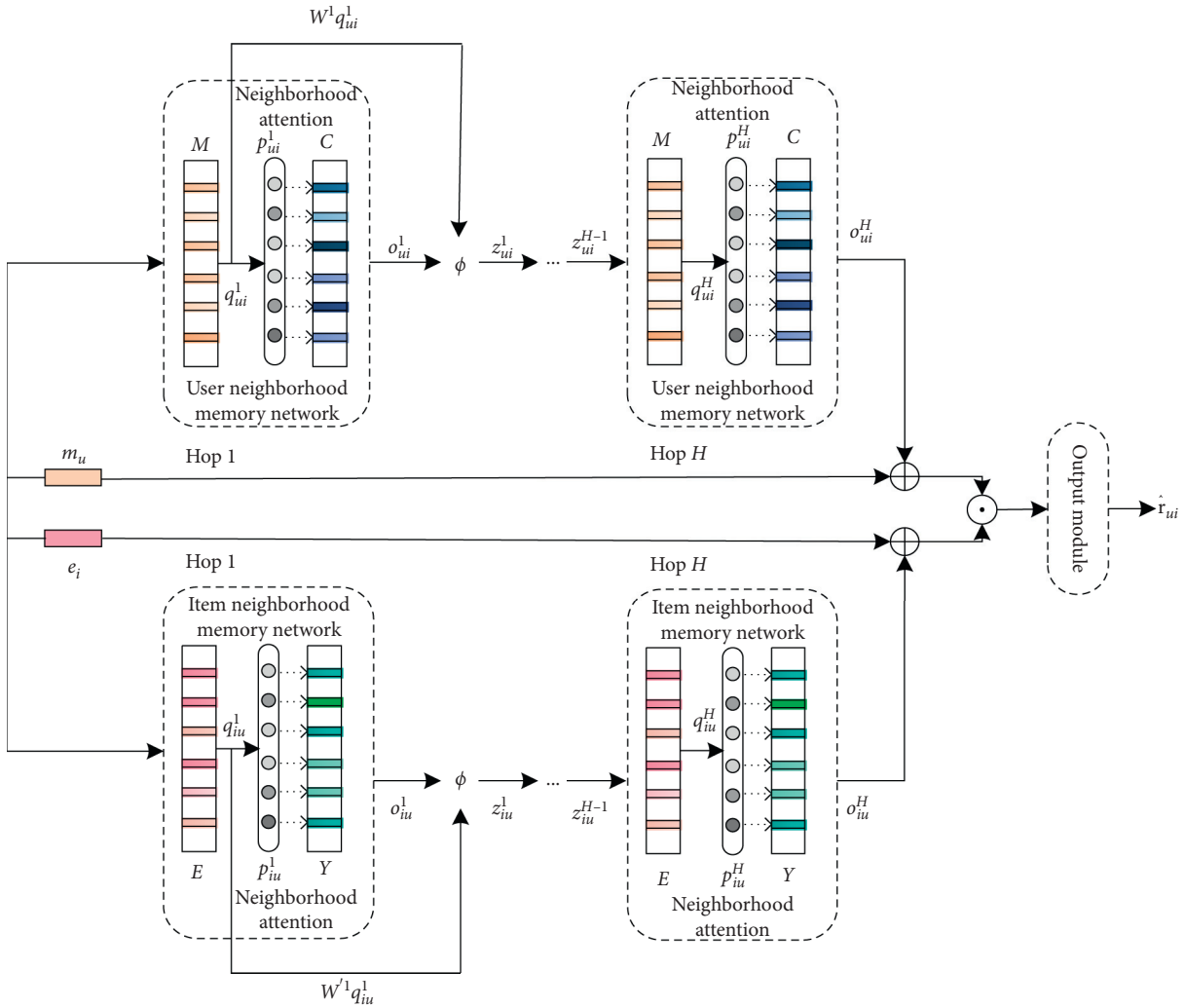

Figure 1: The overall framework of the NAMN.

Table 1: Notations and descriptions.

Notations	Descriptions
M, E	Memory matrix for users and items
C, Y	External memory matrix for users and items
m_u, m_v, e_i, e_j	Embedding vectors for user u, user v, item I, and item j
$N(i)$	User neighborhood set of all users who have accessed item i
$S(u)$	Item neighborhood set of all items that user u has accessed
h, H	The number of hops and the last hop
q_{ui}^h, q_{iu}^h	h-hop user preference vector given item i and item attribute vector given user u
q_{uiv}^h, q_{iuj}^h	The similarity of the user u with user v given item i and the similarity of the item i with item j given user u in h-hop
p_{ui}^h, p_{iu}^h	Attention weights vectors of user neighborhood given item i and attention weights vectors of item neighborhood given user u in h-hop
p_{uiv}^h, p_{iuj}^h	Attention weight of user v given item i and attention weight of item j given user u in h-hop
o_{ui}^h, o_{iu}^h	h-hop user neighborhood representation given item i and item neighborhood representation given user u
z_{ui}^h, z_{iu}^h	h-hop user neighborhood preference representation given item i and item neighborhood attribute representation given user u
W^h, W'^h, U, b^h, v, b	The parameters to be learned in neural networks

neighborhood-based approaches predefine a heuristic weighting function such as a cosine similarity or Pearson correlation, which must consider the number of users or items. Although this problem can be partially alleviated by factorizing the neighborhood, it is still linear in nature. Instead, through the traditional methods that learn a

weighting function over the entire neighborhood, we no longer need to predefine the number of neighbors or weighting functions to consider. For target item i, we compute the attention weights for all the users in the user neighborhood to infer the importance of each user's unique contribution to the user neighborhood. For target user u, we compute the attention weights for all the items in the item neighborhood to obtain the importance of each item's unique contribution to the item neighborhood:

$$p_{uiv}^{1} = \frac{\exp\left(q_{uiv}^{1}\right)}{\sum_{k\in N(i)}\exp\left(q_{uik}^{1}\right)}, \quad \forall v \in N(i),$$
$$p_{iuj}^{1} = \frac{\exp\left(q_{iuj}^{1}\right)}{\sum_{k\in S(u)}\exp\left(q_{iuk}^{1}\right)}, \quad \forall j \in S(u), \tag{3}$$

which produce two distributions over the user neighborhood and the item neighborhood. The attention mechanism allows the model to place higher weights on specific users in the user neighborhood and focus on higher weights on specific items in the item neighborhood, while placing less importance on users and items that may be less similar. We construct the user neighborhood representation and the item neighborhood representation by interpolating the external neighborhood memory with the attention weights:

$$o_{ui}^{1} = \sum_{v\in N(i)} p_{uiv}^{1} c_v, o_{iu}^{1} = \sum_{j\in S(u)} p_{iuj}^{1} y_j, \tag{4}$$

where c_v is another embedding vector for user v, which is called the external memory for user v, denoting thev^{th} column of the user external memory matrix C with the same dimensions as M; y_j is another embedding vector for item j, which is called the external memory for item j, denoting the j^{th} column of the item external memory matrix Y with the same dimensions as C. The external memory allows the storage of long-term information pertaining specifically to each user's and item's role in the neighborhood. In other words, the associative addressing scheme identifies similar users within the user neighborhood or similar items within the item neighborhood, acting as a key to weight the relevant values stored in the external memory matrix C or Y via the attention mechanism. The attention mechanism specifically weights the neighbors according to the target user and item. The output o_{ui}^{1} is the user neighborhood representation given the specific item i, and o_{iu}^{1} is the item neighborhood representation given the specific user u. They are composed of the relations between the specific user, item, and neighborhood.

NAMN captures the similarity between user u and the users who are in the user neighborhood given the specific item i and dynamically assigns the degrees of contribution to the representation of the user neighborhood based on the target item. NAMN also captures the similarity between item i and the items that are in the item neighborhood given the specific user u and dynamically assigns the degrees of contribution to the representation of the item neighborhood based on the target user. NAMN does not need to predefine the number of users in the user neighborhood or the number of items in the item neighborhood, so it has a good generalization capacity. In addition, the attention mechanism simultaneously considers the information of each user in the user neighborhood and each item in the item neighborhood and encodes all the neighborhood information into a single memory slot.

3.3. Multiple Hops. In this section, we extend NAMN to handle an arbitrary number of user and item neighborhood memory networks or hops. Each hop queries the user memory and item memory and is followed by the attention mechanism to obtain the next user neighborhood representative vector and item neighborhood representative vector. The first hop may barely capture the higher-order complex information. Starting from the second hop, the model begins to take into consideration the information of the user neighborhood and the item neighborhood, guiding the search for the representation of the user preferences and the item attributes, respectively. Each additional hop repeats this step considering the previous hop's newly acquired information, before producing the final neighborhood representation. In other words, the model has the chance to look back and reconsider the most similar users and items and to infer more precise information for the user neighborhood and the item neighborhood. More specifically, multiple memory modules are stacked together by taking the output from the h hop as input to the $h+1$ hop. We apply two nonlinear projections between different hops:

$$z_{ui}^{h} = \phi\left(W^{h} q_{ui}^{h} + o_{ui}^{h} + b^{h}\right), \tag{5}$$

$$z_{iu}^{h} = \phi\left(W'^{h} q_{iu}^{h} + o_{iu}^{h} + b'^{h}\right). \tag{6}$$

In formula (5), W^{h} is a weight matrix mapping the user preference query q_{ui}^{h} to a latent space, coupled with the user neighborhood information from the previous hop, followed by a nonlinear activation function $\phi(\cdot)$ to obtain the new user preference representation z_{ui}^{h}. In formula (6), W'^{h} is another weight matrix mapping the item attribute query q_{iu}^{h} to a latent space, coupled with the item neighborhood information from the previous hop, followed by a nonlinear activation function $\phi(\cdot)$ to obtain the new item attribute representation z_{iu}^{h}. Intuitively, multiple hops provide additional information that allows inference of more precise user neighborhood information and item neighborhood information. Through the above operations, the vector representation of user preference information and item attribute information are updated and then it reconsiders the relations between them and the user neighborhood and item neighborhood:

$$q_{uiv}^{h+1} = \left(z_{ui}^{h}\right)^{T} m_v, \quad \forall v \in N(i), \tag{7}$$

$$q_{iuj}^{h+1} = \left(z_{iu}^{h}\right)^{T} e_j, \quad \forall j \in S(u). \tag{8}$$

Formula (7) applies the dot product of the newly formed user preference information with the user memory in the user neighborhood and obtains the similarity between the

target user and the users in the user neighborhood. Then, via an adaptive attention mechanism, it produces an updated user neighborhood representation. Formula (8) applies the dot product of the newly formed item attribute information with the item memory in the item neighborhood and obtains the similarity between the target item and the items in the item neighborhood. Then, via an adaptive attention mechanism, it produces an updated item neighborhood representation. The abovementioned process is repeated for each hop, yielding an iterative refinement. The output module will receive the user neighborhood representation in the user neighborhood memory network and the item neighborhood representation in the item neighborhood memory network from the last H hop to produce the final recommendation.

3.4. Output Module. As mentioned above, traditional neighborhood-based models identify the local structure by analyzing similarities between users or items within the neighborhood, while latent factor models capture the global structure by transforming both users and items to the same latent factor space [3]. Hence, we consider the user neighborhood representation and item neighborhood representation to identify localized user-item interactions and the user and item memories to identify the global user-item relations. Existing methods lack the nonlinear interaction between the local structure and the global structure that barely allows the capture of deeper relations [27]. For a given user u and item i the ranking score can be formulated as

$$\hat{r}_{ui} = v^T \phi\left(U\left(\left(m_u + o_{ui}^H\right) \odot \left(e_i + o_{iu}^H\right)\right) + b\right), \tag{9}$$

where $\odot$ is the elementwise product; $U \in R^{2d \times 2d}$ and $v, b \in R^{2d}$ are the parameters to be learned; H is the number of the last hop. In collaborative memory network (CMN), the element product was first applied to the user and item memories followed by a linear projection and subsequently introduced a skip-connection combined with the user neighborhood representation that followed by another linear projection [25, 26]. In NAMN, we first introduce two skip-connections that combine user memory m_u and user neighborhood representation o_{ui}^H and then combine item memory e_i and item neighborhood representation o_{iu}^H. Skip-connections can reduce the longest path from the output to input, which encourage the flow of information and ease the learning process [26]. In this way, the model can better correlate the specific user and item memories with the ranking score. Subsequently, we apply the elementwise product between the two combinations, and then the result of elementwise product is projected to a latent space with U, followed by a nonlinear activation function $\phi(\cdot)$. Through extensive experiments, we found the rectified linear unit (ReLU) $\phi(x) = \max(0, x)$ to work best.

3.5. Parameter Optimization. In NAMN, we intend to study collaborative filtering based on implicit feedback, which is more pervasive in the real world and can be collected automatically (e.g., likes and clicks). The rating matrix from the user's implicit feedback contains a 1 if the interaction is observed and a 0 otherwise. We make for a pairwise assumption that the target user u prefers the observed item i^+over the unobserved or negative item i^-. In reality, a value of 1 does not mean user u actually likes item i^+. Similarly, user u may not dislike item i^-; it may be that user u is not aware of item i^-. We uniformly sample a ratio of positive items to negative items to form triplet preferences (u, i^+, i^-), which we further investigate in Section 4.5. The loss function of our model is the Bayesian personalized ranking (BPR) optimization criterion which approximates AUC (area under the ROC curve):

$$\tau = -\sum_{(u,i^+,i^-)} \log \delta\left(\hat{r}_{ui^+} - \hat{r}_{ui^-}\right), \tag{10}$$

where $\delta(x) = 1/(1 + \exp(-x))$ is the logistic sigmoid function. Since the entire architecture is differentiable, NAMN can be efficiently trained with the backpropagation algorithm. Similar to [23, 27], we utilize layerwise weight tying sharing all embedding matrices across hops to reduce the number of parameters.

3.6. Computational Complexity. The computational complexity for a forward pass through NAMN for a given user-item pair is $O(d|N(i)| + d|S(u)| + d^2 + d)$, where $|N(i)|$ and $|S(u)|$ represent the size of the user neighborhood set for item i and the item neighborhood set for user u respectively and d denotes the size of each memory cell. The first two terms are the costs for computing the user preference vector and the item attribute vector and the latter terms are the costs of the output module. Each additional hop introduces $O(d|N(i)| + d|S(u)| + d^2)$ complexity. NAMN calculates two forward passes during training, one for the observed positive term and another for the unobserved negative term. Parameters can be updated via backpropagation with the same complexity. In public datasets, $|N(i)|$ and $|S(u)|$ are usually slightly larger than or comparable to d. Therefore, the primary complexity for evaluating NAMN is $O(d|N(i)| + d|S(u)|)$. The cost is reasonable since other deep learning methods such as CMN [25] have computational complexity $O(d|N(i)| + d^2 + d)$. NAMN requires only extra computation of the similarity with the target item's neighbors and $|S(u)|$ is often less than or comparable to $|N(i)|$. Thus, the training in NAMN is also quite efficient.

Recommendation can be performed by computing the ranking score (equation (9)) for a given user-item pair with a single pass through the network. The top-N items with the highest score are recommended to the user. The computational complexity during testing is the same with that of the single forward pass during training.

4. Experimental

In this section, we evaluate NAMN on three real-world datasets: Epinions, citeulike-a, and Pinterest. We first introduce the datasets, evaluation metrics, baselines, and settings and then present the baseline comparisons and

parameter sensitivity. Last, we discuss the effects of the neighborhood memory network.

4.1. Datasets. In our experiments, we use three publicly accessible datasets, i.e., Epinions, citeulike-a, and Pinterest, to evaluate the effectiveness of our model. The details of the three datasets are as follows:

(i) *Epinions*. The first dataset from Epinions allows users to share product feedback in the form of explicit ratings and reviews. If the user has rated the product (item), we convert the explicit ratings to implicit feedback as 1 and 0 otherwise.

(ii) *Citeulike-a*. The second dataset is collected from a literature management system called CiteULike, which provides users with a digital directory to store and share academic papers. If the paper (item) is saved in the user's online catalog, the user preferences are encoded as 1 and 0 otherwise.

(iii) *Pinterest*. The third dataset is collected from Pinterest, the largest photo sharing site in the world, which allows users to save an image to their board. Each interaction denotes whether the user has saved the image (item) to his/her own board.

The statistical details of the three datasets are presented in Table 2.

4.2. Evaluation Metrics. We use the leave-one-out evaluation method to evaluate the performance of our proposed model, which has been widely used in the literature [10, 13, 25]. In terms of experimental setup, we follow a common strategy, which randomly samples 100 negative items and one positive item to form the test set for each user. For each of the remaining positive examples, we randomly sample 4 negative items for training. To alleviate the cold-start setting, if the user has only one interaction, we put it in the training set. We rank the positive item among the 100 negative items, the performance of a ranked list is judged by Hit Ratio (HR) and normalized discounted cumulative gain (NDCG). HR measures whether the positive item is present in the top N, and NDCG accounts for the position of the positive item hit by penalizing the score for ranking the positive item lower in the list.

4.3. Baselines and Settings. We compare the proposed NAMN with seven state-of-the-art baselines, which are selected from three kinds of CF approaches and the deep learning-based models.

(i) BPR [28] is a traditional pairwise matrix factorization for implicit feedback.

(ii) GMF [13] is a traditional latent factor model. We use the ReLU activation function to generalize MF to a nonlinear setting and optimize the BPR loss function.

(iii) FISM [29] is a neighborhood-based CF model for top-N recommender systems that learns the item-item similarity matrix as the product of two low dimensional matrices by optimizing the BPR loss function.

(iv) KNN [30] is a neighborhood-based CF approach computing the cosine item-item similarity to provide recommendations.

(v) SVD++ [3] is a hybrid CF model that smoothly merges the latent factor model and the neighborhood-base method.

(vi) NeuMF [13] is a deep learning-based model combining matrix factorization and a multilayer perceptron model.

(vii) CMN [25] is a competitive hybrid deep learning-based model which fuses the global structure of the latent factor model with the local structure of the user neighborhood or memory.

We leave out the comparisons with the baselines that utilize additional information (e.g., content and contextual) since our objective is to study collaborative filtering based on implicit feedback. We randomly select one interaction for each user from the training set to form the validation set and tune all hyperparameters on it. We set the hop number to 2 and the number of negative samples to 4, and the embedding size of the user and item memory is set to 120. The decay rate is set to 0.9, both the regularization term and the learning rate are set to 0.001. The batch size of Epinions and citeulike-a is set to 128 and 256 for Pinterest. Since the initialization of the deep neural networks is crucial, NAMN initializes the user and item memory from the generalized matrix factorization model [13]. In Section 4.5, we will further explore the effects of some hyperparameters.

4.4. Baseline Comparison. Table 3 shows the experimental results of NAMN, along with the baselines, in terms of HR and NDCG with cut offs at 5 and 10 on the Epinions, citeulike-a, and Pinterest datasets, respectively. We denote a NAMN with two hops as "NAMN-2" and "CMN-2" stand for a CMN with two hops. Clearly, NAMN obtains the best performance across all benchmark datasets and metrics. SVD++ performs better than the latent factor model BPR and the neighborhood-based method FISM on the citeulike-a dataset, which demonstrates the superiority of hybrid model. The nonlinear generalization of the latent factor model GMF outperforms the linear BPR on the citeulike-a dataset, which reveals the effectiveness of the nonlinear model. The linear decomposition of the item-item similarity matrix in FISM performs unsatisfactorily in general. KNN shows poorest performance with each other across all metrics and cut offs on Epinions, which is mainly due to the restrictive ability to handle sparse data. The Pinterest dataset contains fewer items than the citeulike-a dataset leading to poor performance from the item-based FISM and KNN methods due to the presence of only a few neighbors. SVD++ shows competitive performance across all datasets but lacks the full expressiveness of nonlinearity found in the deep learning-based models, NeuMF, and CMN. CMN can be viewed as a modified NeuMF, by replacing the original multilayer perceptron with a user memory network

Table 2: Dataset statistics.

Datasets	Interactions	Users	Items	Density (%)
Epinions	664,823	40,163	139,738	0.01
Citeulike-a	204,987	5,551	16,980	0.22
Pinterest	1,500,809	55,187	9,916	0.27

Table 3: Experimental results for different methods on the Epinions, citeulike-a, and Pinterest datasets.

	Epinions				Citeulike-a				Pinterest			
	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10	HR@5	HR@10	NDCG@5	NDCG@10
BPR	0.5584	0.6659	0.4334	0.4683	0.6547	0.8083	0.4858	0.5357	0.6936	0.8674	0.4912	0.4912
GMF	0.5365	0.6562	0.4016	0.4404	0.7271	0.8326	0.5689	0.6034	0.6726	0.8505	0.4737	0.5316
FISM	0.5542	0.6717	0.4192	0.4573	0.6727	0.8072	0.5106	0.5545	0.6783	0.8654	0.4658	0.5268
KNN	0.1549	0.1555	0.1433	0.1435	0.6990	0.7348	0.5789	0.5909	0.5738	0.8376	0.3450	0.4310
SVD++	0.5628	0.6754	0.4112	0.4477	0.6952	0.8199	0.5244	0.5649	0.6951	0.8684	0.4796	0.5362
NeuMF	0.5500	0.6660	0.4214	0.4590	0.7629	0.8647	0.5985	0.6316	0.7041	0.8732	0.4978	0.5530
CMN-2	0.6017	0.7007	0.4724	0.5045	0.7959	0.8921	0.6185	0.6500	0.7267	0.8904	0.5180	0.5714
NAMN-2	**0.6230**	**0.7122**	**0.4913**	**0.5234**	**0.8233**	**0.9101**	**0.6576**	**0.6867**	**0.7470**	**0.8966**	**0.5432**	**0.5918**

component. CMN achieves better performance than NeuMF, further demonstrating the effectiveness of the memory network component, which iteratively updates the internal user neighborhood state to identify complex interactions. NAMN performs best among all the methods in all datasets, demonstrating the superiority of combining a user neighborhood memory network with an item neighborhood memory network to cope with the collaborative filtering task based on implicit feedback.

4.5. Parameter Sensitivity. In this subsection, we conduct an empirical study to investigate the effect of varying the size of embeddings, the number of negative samples, and hops for HR@10 and NDCG@10 on the citeulike-a dataset, which have similar trends on the Epinions and Pinterest datasets.

4.5.1. Size of Embeddings in 2-Hop. We observe from Figure 2 that embedding sizes range from 20 to 200, while other parameters are kept fixed. The results of HR and NDCG show the same trend that, with the increase of embedding size, the performance is boosted initially since larger dimensions can encode more useful information. Both HR and NDCG show steady improvement as embedding sizes increase, with the exception of HR, where an embedding size of 140 shows a performance drop due to the nonconvex nature of neural networks.

4.5.2. Negative Samples Number in 2-Hop. We show the performance variances of the NAMN method with the negative samples from 2–10 in Figure 3. We exclude the results of 1 negative sample since NAMN was unable to distinguish between positive and negative samples, leading to a random performance. As shown, just two negative samples for each positive sample are insufficient to achieve a competitive performance, and sampling more negative instances is beneficial for recommendations. For both metrics, the best sampling numbers are approximately 4 to 6 instances. We also find that when a negative sample number is larger than 6, the performance of NAMN starts to drop. This demonstrates that setting the negative sample number too aggressively may adversely affect the performance of the model.

4.5.3. Hop Number. We also illustrate the effect of changing the maximal hop number H for HR@10 and NDCG@10 on the citeulike-a dataset in Table 4. As shown, our proposed model has achieved the best performance when H is 1 or 2. More specifically, the performance for HR@10 is best when H is 1, and it is very close to the result when H is 2, while the performance for NDCG@10 is best when H is 2. Hence, two hops can explore higher-order user-item neighborhood relations, and too large of an H brings much more noise than useful information.

4.6. Effects of Neighborhood Memory Network. In this subsection, we further explore the effect of individual neighborhood memory network components for HR@10 and NDCG@10 on the citeulike-a dataset, which have similar trends to the Epinions and Pinterest datasets. We denote a NAMN without an item neighborhood memory network as "NAMN-user," and "NAMN-item" stands for a NAMN without a user neighborhood memory network. In Table 5, the results for "NAMN-user" uniformly perform worse than NAMN, hinting at the effectiveness of the item neighborhood memory network. Another variation, "NAMN-item," generally performs worse than NAMN, revealing the effectiveness of the user neighborhood memory network. Furthermore, "NAMN-user" shows improvements over "NAMN-item." It appears that the user neighborhood memory network contains more useful information to explore users' potential preferences and hierarchical interests. Generally, NAMN requires a combination from both the user neighborhood

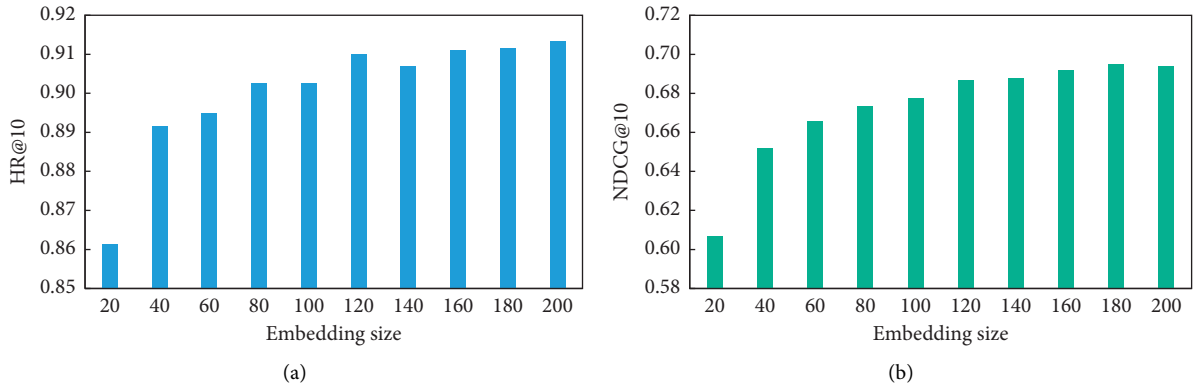

Figure 2: Experimental results of varying embedding size on citeulike-a for NAMN-2.

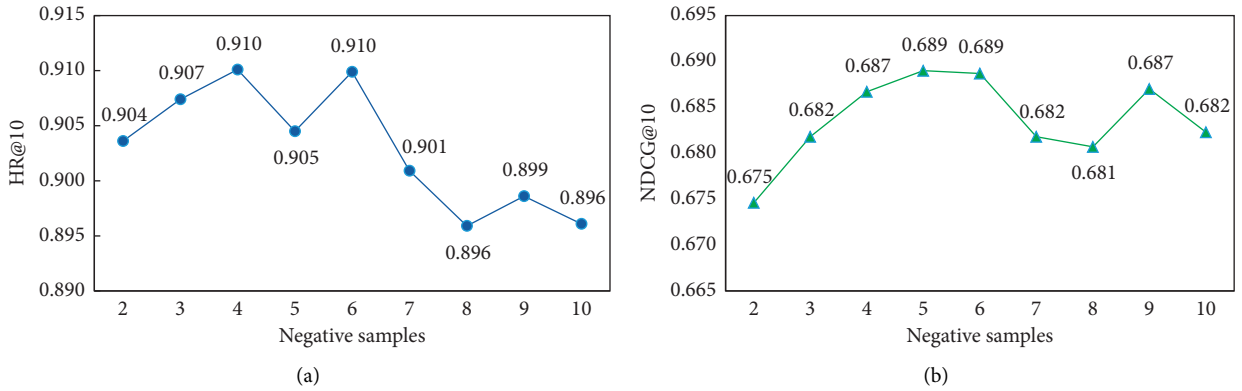

Figure 3: Experimental results of varying negative samples on citeulike-a for NAMN-2.

Table 4: Experimental results for different hops on the citeulike-a.

Hop number H	1	2	3	4
HR@10	**0.9103**	0.9101	0.9031	0.9074
NDCG@10	0.6845	**0.6867**	0.6820	0.6865

Table 5: NAMN variants without item or user neighborhood memory network (NAMN-user or NAMN-item).

	Citeulike-a			
	HR@5	HR@10	NDCG@5	NDCG@10
NAMN	0.8233	0.9101	0.6576	0.6867
NAMN-user	0.8151	0.8938	0.6536	0.6793
NAMN-item	0.8108	0.8938	0.6488	0.6761

memory network and the item neighborhood memory network to obtain the optimal performance.

5. Conclusion

In this paper, we propose a novel deep learning recommendation model NAMN, which takes the advantages of memory networks to deal with collaborative filtering based on implicit feedback. NAMN overcomes the limitations of existing methods, which only directly profile both users and items, by designing a user neighborhood memory network and an item neighborhood memory network to capture higher-order complex users' neighborhood relations and items' neighborhood relations, respectively. Extensive experiments on three datasets demonstrate that our proposed model outperforms strong baselines and reveal the effectiveness of both the user neighborhood memory network and the item neighborhood memory network.

In future work, we plan to extend NAMN to incorporate other auxiliary information, such as user reviews, temporal signals, and knowledge graphs to better explore users' potential interests and items' potential attributes.

Data Availability

Previously reported citeulike-a dataset and Pinterest dataset were used to support this study and are available at http://github.com/tebesu/CollaborativeMemoryNetwork. These prior studies (and datasets) are cited at relevant places within the text as reference [25].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (nos. 62066010, 61966009, U1811264, and U1711263), the Natural Science Foundation of Guangxi Province (no. 2020GXNSFAA159055), the Guangxi Innovation-Driven Development Grand Project (no. AA17202024), and the Innovation Project of GUET Graduate Education (no. 2020YCXS047).

References

[1] H. Zhang, F. Shen, W. Liu, X. He, H. Luan, and T.-S. Chua, "Discrete collaborative filtering," in *Proceedings of the SIGIR*, Pisa, Italy, July 2016.

[2] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[3] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the SIGKDD*, Las Vegas, NV, USA, March 2008.

[4] S. Rendle, "Factorization machines," in *Proceedings of the 2010 IEEE International Conference on Data Mining*, IEEE, Sydney, Australia, December 2010.

[5] G. Zhou, N. Mou, Y. Fan et al., "Deep interest evolution network for click-through rate prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 5941–5948, 2019.

[6] B. Liu, R. Tang, Y. Chen, J. Yu, H. Guo, and Y. Zhang, "Feature generation by convolutional neural network for click-through rate prediction," in *Proceedings of the 28th International Conference of World Wide Web (WWW)*, pp. 1119–1129, Geneva, Switzerland, April 2019.

[7] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: deep knowledge-aware network for news recommendation," in *Proceedings of the WWW 2018*, Lyon, France, April 2018.

[8] H. Wang, F. Zhang, J. Wang et al., "RippleNet: propagating user preferences on the knowledge graph for recommender systems," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM 2018)*, ACM, Torino, Italy, October 2018.

[9] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," 2017, https://arxiv.org/abs/1706.03762.

[10] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proceedings of the SIGIR*, pp. 549–558, Pisa, Italy, July 2016.

[11] H. Liu, T. Li, R. Hu, Y. Fu, J. Gu, and H. Xiong, "Joint representation learning for multi-modal transportation recommendation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1036–1043, 2019.

[12] H. Guo, R. Tang, Y. Ye et al., "DeepFM: a factorization-machine based neural network for CTR prediction," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, Melbourne, Australia, August 2017.

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

[14] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proceedings of the SIGIR*, pp. 355–364, Tokyo, Japan, August 2017.

[15] X. Xin, B. Chen, X. He et al., "CFM: convolutional factorization machines for context-aware recommendation," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence IJCAI-19*, Macao, China, August 2019.

[16] X. Zhou, D. Liu, J. Lian et al., "Collaborative metric learning with memory network for multi-relational recommender systems," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, Macao, China, August 2019.

[17] C. Chen, M. Zhang, Y. Liu et al., "Social attentional memory network: modeling aspect- and friend-level differences in recommendation," in *Proceedings of the Twelfth ACM International Conference*, ACM, Melbourne, Australia, February 2019.

[18] Z. Yu, J. Lian, A. Mahmoody et al., "Adaptive user modeling with long and short-term preferences for personalized recommendation," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence IJCAI-19*, Macao, China, August 2019.

[19] Y. Gong and Q. Zhang, "Hashtag recommendation using attention-based convolutional neural network," in *Proceedings of the IJCAI*, New York, NY, USA, July 2016.

[20] W. Zaremba and I. Sutskever, "Learning to execute," 2014, https://arxiv.org/abs/1410.4615.

[21] J. Weston, S. Chopra, and A. Bordes, "Memory networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 2015.

[22] S. Sukhbaatar, J. Weston, R. Fergus et al., "End-to-end memory networks," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2440–2448, Montreal, Canada, December 2015.

[23] H. Huang, Q. Zhang, X. Huang, H. Huang, Q. Zhang, and X. Huang, "Mention recommendation for twitter with end-to-end memory network," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 1872–1878, Melbourne, Australia, August 2017.

[24] X. Chen, H. Xu, Y. Zhang et al., "Sequential recommendation with user memory networks," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 108–116, ACM, Los Angeles, CA, USA, February 2018.

[25] T. Ebesu, B. Shen, and Yi Fang, "Collaborative memory network for recommendation systems," in *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 515–524, ACM, Arbor, MI, USA, July 2018.

[26] T. A. Ebesu, *Deep learning for recommender systems*, vol. 22, Santa Clara University, Santa Clara, CA, USA, 2019, Engineering Ph.D. theses.

[27] S. Zhang, L. Yao, and X. Xu, "AutoSVD++: an effeicient hybrid collaborative filtering model via contractive auto-encoders," *ACM SIGIR Forum*, vol. 51, pp. 957–960, 2017.

[28] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-thieme, "BPR: bayesian personalized ranking from implicit feedback," in *Proceedings of the UAI*, Montreal, Canada, June 2009.

[29] SantoshKabbur and X. Ning, "GeorgeKarypis:FISM: factored item similarity models for top-N recommender systems," in *Proceedings of the KDD*, pp. 659–666, Chicago, IL, USA, August 2013.

[30] F. Ricci, L. Rokach, and B. Shapira, *Introduction to Recommender Systems Handbook*, Springer, Berlin, Germany, 2011.