Coupled Linear and Deep Nonlinear Method for Meetup Service Recommendation

Shuai Zhang[1](✉), Lina Yao[1], Xiaodong Ning[1], Chaoran Huang[1], Xiwei Xu[2], and Shiyan Ou[3]

[1] School of Computer Science and Engineering, University of New South Wales, Kensington, Australia
{shuai.zhang,xiaodong.ning}@student.unsw.edu.au,
{lina.yao,chaoran.huang}@unsw.edu.au

[2] Data61, Commonwealth Scientific and Industrial Research Organisation, Canberra, Australia
XiWei.Xu@data61.csiro.au

[3] School of Information Management, Nanjing University, Nanjing, China
oushiyan@nju.edu.cn

Abstract. Meetup brings people with similar interests together to do things that matter to them. For example, it provides a platform for getting people who love hiking, coding, running marathons, learning foreign languages together so that they can help, teach and learn from each other. Thanks to the development of web and mobile technologies, organizing these Meetup groups has become much more easily than before. Meetup has become an ideal tool for enriching one's social life. In this paper, we proposed a coupled linear and deep nonlinear method for Meetup services recommendation. Our method considers both historical user item interactions and group features by combining linear model with deep neural networks. In addition, we designed a pairwise training algorithm with dynamic negative sampling technique to further enhance the model performance. Experiments on two real-world datasets show that our approach outperforms the compared state-of-the-art methods by a large margin.

Keywords: Recommender systems · Deep learning
Service recommendation

1 Introduction

Meetup[1] is a social networking website for organizing local offline group meetings for people with similar interests. Thousands of Meetup groups, such as fitness group, career and networking group, photography group, hiking group, etc., are there for us to participate in. It provides a desirable approach to enrich our

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

social life. For example, we can find a fitness partner in the Meetup group to support and encourage each other, or get someone to mentor us on photography. These Meetup groups provide us with a good way to explore the things we are interested in, meet new friends, broaden our social circle and even change our careers.

With so many Meetup group choices being available, a good recommender model can save our time in finding interesting Meetup groups, attracting more group members and making them more active. To this end, we propose to explore user historical interactions as well as group features to better match user interests with Meetup groups. The main contributions of this work are summarized as follows:

- A coupled linear and deep nonlinear recommendation model is proposed to integrate both historical interactions as well as item side information. It can capture both user's historical preferences and item characteristics.
- We designed a pairwise learning algorithm for the proposed approach. To further improve the recommendation quality, we also adopted a dynamic negative sampling approach to conduct negative sampling more effectively.
- We did extensive experiments on two large-scale datasets and demonstrated the superior performances of our approach over state-of-the-art baselines.

The reminder of this paper is structured as follows. In the next section, we will introduce the research problem we aim to address. Section 3 introduces the proposed approach. Section 4 shows the experimental setup and results. Section 5 introduces the related work and Sect. 6 concludes this paper.

2 Problem Formulation

Assuming that there are N items and M users, we have an interaction matrix $X \in \mathcal{R}^{M \times N}$, and most entries of X are unobserved. Let X_{ui} denote the preference of user u to item i, X_{u*} denote the u^{th} row of the interaction matrix. For Meetup recommendation, there are only binary implicit feedback available and it can be viewed as a one-class recommendation problem [12]. The entries of X are defined as follows:

$$X_{ui} = \begin{cases} 1, & \text{if interaction } {<}u, i{>} \text{ is observed} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

The goal of the recommendation is to predict ranking scores for unobserved entires given the observed interactions, and then generate a personalized ordered list of items for each user based on the predicted scores. For a clear presentation, Table 1 summarizes the notations and denotations used in this paper.

3 Proposed Methodology

In this section, we will introduce the proposed methodology in detail. Our model combines a linear part to capture the user historical interactions and a nonlinear component to incorporate the abundant side information.

Table 1. Notations and denotations

Notations	Descriptions
M, N	Number of users and items
X, X_{u*}	Interaction matrix, and the u^{th} row
s_i	Side information of item i (or item content information)
$A \in \mathcal{R}^{N\times N}$	Sparse aggregation co-efficient matrix
W_*, b_*	Weights and biases of neural networks
$P \in \mathcal{R}^{M\times k}$	User latent factor
k	Dimension of the output of neural networks and latent factor size
Y_{ui}	Predicted ranking score of item i for user u
η, λ, τ, t	Learning rate, regularization rate, scaling factor, negative sample size

3.1 Coupled Linear and Deep Nonlinear Model

Sparse linear model has demonstrated to be effective for top-n recommendations [20]. However, this model does not consider any side information. In recent years, deep learning has demonstrated to be very suitable for feature representation learning [1]. Therefore, we propose using deep neural networks to learn low dimensional feature embeddings from raw features. Since both usage history and item properties are critical for uncovering user's real demands and interests, here, we design a hybrid model which couples sparse linear model with deep neural network for better service recommendation. The former (sparse linear model) is used to learn user's interaction patterns, while the latter (deep neural network) aims to understand the content of items.

Formally, let $A \in \mathcal{R}^{N\times N}$ denote a sparse aggregation co-efficient matrix. The ranking score of the linear part is calculated by

$$Y^I_{ui} = X_{u*} \cdot A_{*i} \tag{2}$$

where X_{u*} is the u^{th} row the interaction matrix, and it is constructed from training set, so there is no leakage of the test data. Equation (2) is very similar to matrix factorization. We can view X_{u*} as the user latent factor and A_{*i} as the item latent factor. Nevertheless, X_{u*} is a known vector, and A is a sparse co-efficient matrix needed to be optimized. Moreover, A is reminiscent of the similarity matrix in item-based neighborhood collaborative filtering [15], but it is determined by minimizing a predefined loss rather than being calculated with Cosine or Jaccard similarities from the interaction matrix. Due to the sparse nature of X_{u*}, some constraints such as sparsity and non-negativity are put on the co-efficient matrix A. More details will be introduced in the following text.

Another important component of our model is a deep neural network, which is used to integrate side information of items to further enhance the recommendation performance. Let s_i denote the side information of item i. We first feed it

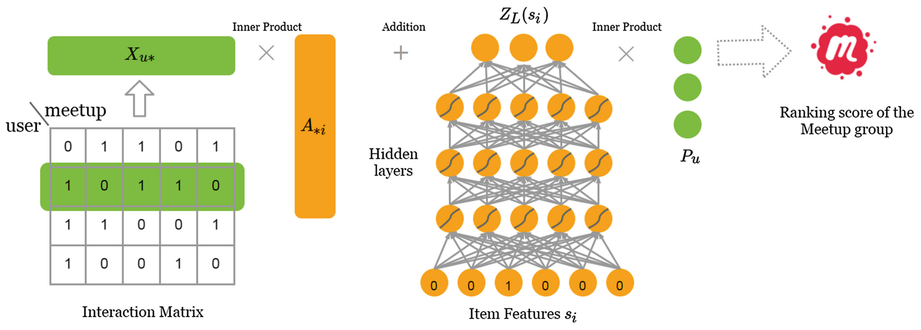


Fig. 1. Illustration of the coupled linear and deep nonlinear method. It consists of two components: a linear component used to learn patterns from historical interactions; and a deep neural network used to capture the item features.

into a multi-layered neural network and get the high level dense representations. Formally, the definition of the multi-layered neural network is as follows:

$$\begin{aligned} z_1(s_i) &= \sigma_1(W_1 s_i + b_1) \\ z_2(s_i) &= \sigma_2(W_2 z_1(s_i) + b_2) \\ &\cdots \\ z_L(s_i) &= \sigma_L(W_L z_{L-1}(s_i) + b_L) \end{aligned}$$

where L denotes the number of layers, W_l and b_l denote the weight matrix and bias vector of the l^{th} layer. σ_l is the activation function which could be sigmoid, hyperbolic tangent (tanh) or Rectifier (ReLU). With this nonlinear transformation, we manage to capture the complex and intricate data structure of item side information. Let k denote the dimension of the output, so $Z_L(s_i)$ is a k-dimensional vector. To integrate this neural network into the recommendation model, we define a user latent factor $P \in \mathcal{R}^{M \times k}$, and then model the user and item interactions with inner product:

$$Y_{ui}^{II} = P_u \cdot Z_L(s_i) \tag{3}$$

Finally, we simply add the former two scoring results and get the final predicted ranking score.

$$Y_{ui} = Y_{ui}^{I} + Y_{ui}^{II} \tag{4}$$

Figure 1 illustrates the structure of the proposed methodology. The left part is the linear component and the right part is the deep neural network.

3.2 Pairwise Training Algorithm

To train the above model in a pointwise manner is computationally intensive. To accelerate the training process, here, we propose learning this model with a pairwise algorithm. We adopt a logarithm function which has a scaling factor to weight the difference between positive and negative samples. Formally, the loss function of our model is defined as follows:

$$\mathcal{L}(\theta) = \sum_{(u,i^+,i^-)} log(1 + exp(-\tau\Delta)) + \lambda\Omega(\theta) \tag{5}$$

where Δ is formulated as $\Delta = (Y_{ui^+} - Y_{ui^-})$, i^+ is the Meetup group that user u joined in and i^- is a negative item that the user has not interacted with. τ is a scaling factor to weight Δ. As indicated in Fig. 2(a), τ can impact the convergence speed as it puts significant influence on the slope of the loss function. θ is the model parameters including A, P, and neural network parameters W_* and b_*.

The regularization terms are critical for the model performance. To ensure the sparse properties of co-efficient A, we put both ℓ_1 and Frobenius norm constraints on it. For other parameters, we find that Frobenius norm is sufficient. Thus, we have:

$$\Omega(\theta) = \parallel A \parallel_1 + \parallel A \parallel_F^2 + \parallel P \parallel_F^2 + \parallel W \parallel_F^2 \tag{6}$$

In addition, we set the diagonal of A to zero and clip the value of A after each iteration to ensure $A \geq 0$.

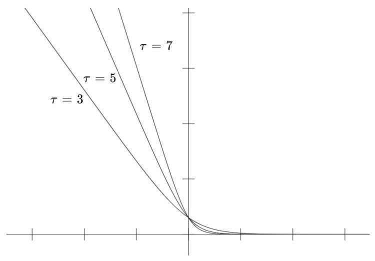
Procedure 1. Training Procedure of the Proposed Methodology

Input: X, s_i, k, learning rate η, λ, batch size, number of neurons for each layer
Output: A, P, W, b

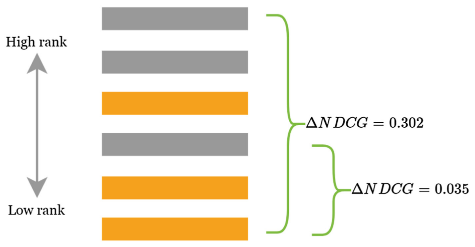
```
 1: procedure INITIALIZATION
 2:     Initialize $A$, $P$, $W$, $b$ with random normal distribution
 3: end procedure
 4: procedure MODEL LEARNING
 5:     repeat
 6:         for all pairs $(u, i)$ in current batch do
 7:             Sample $t$ items from the negative candidates of user $u$
 8:             Calculate the ranking score with equation (4) with current parameters
 9:             Choose the highest ranked item from the sampled $t$ items as the negative sample
10:             Minimize equation (5) with Adam algorithm
11:             Set the diagonal of $A$ to zero and clip $A$ to satisfy $A \geq 0$
12:         end for
13:     until convergence
14: end procedure
```

3.3 Dynamic Negative Sampling

We usually conduct random sampling to sample negative items for each <user, positive item> pair. However, this sampling strategy will not lead to optimal



(a)



(b)

Fig. 2. (a): The influence of parameter τ for the pairwise logarithm function; (b): Example of NDCG differences by exchanging the positions of observed (yellow) and unobserved (gray) items (Best viewed in color).

solutions. One reason is that it cannot guarantee to rank all negative items lower than positive items, while higher ranked negative items will hurt the ranking performance of the current model [31]. Figure 2(b) illustrates this point with an example (taken from [31]). We find that if we exchange the positions of the sixth item (observed) with the first item (unobserved), we increase the NDCG by 0.302. While the NDCG increase is only 0.035 if we exchange the sixth item with the fourth item. Therefore, it is better to rank all unobserved items lower than observed items.

This idea is initially designed for Bayesian personalized ranking model [21]. Here, we find that this assumption is also reasonable for our approach. Therefore, we propose applying the dynamic negative sampling method to our model, the sampling strategy is: in each epoch, we randomly sampled t items from negative candidates for each <user, positive item> pair, and calculate their ranking scores, and then treat the item with highest rank as the negative sample. Procedure 1 summarizes the training process of the proposed model.

4 Experiments

In this section, we conduct experiments on two Meetup datasets and compare our approach with several state-of-the-art baselines.

4.1 Datasets Description

These two datasets are collected by Hsieh et al. [11]. We also crawled the Meetup features from the Meetup websites. After removing Meetup groups without content information and users who interacted with less than 20 Meetup groups, we get two subsets: Meetup San Francisco and Meetup New York city. Detail statistics of the two datasets are summarized in Table 2. These two datasets contain thousands of Meetup groups and regular users from San Francisco and New York city. There are 33 categories of the Meetup groups which spread across most aspects of daily life, including: career & business, education & learning, outdoors & adventure, singles, new age & spirituality, support, games, hobbies & crafts, socializing, paranormal, cars & motorcycles, language & ethnic identity, parents & family, photography, music, sports & recreation, alternative lifestyle, tech, fine arts & culture, LGBT, movements & politics, religion & beliefs, pets & animals, fashion & beauty, fitness, food & drink, writing, sci-fi & fantasy, movies & film, book clubs, health & wellbeing, community & environment, dancing. The category distributions of two cities are shown in Fig. 3.

Table 2. Statistics of datasets meetup San Francisco and New York City.

Datasets	Items #	Users #	Interactions #	Density (%)
Meetup San Francisco	3424	48369	1720017	1.039
Meetup New York City	4753	35889	1055725	0.619

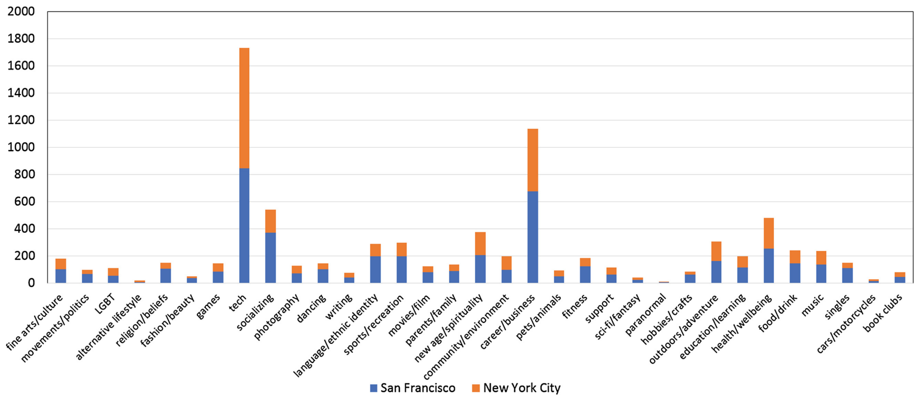


Fig. 3. Statistics of Meetup category distribution for San Francisco and New York City.

4.2 Evaluation Metrics

To evaluate the recommendation accuracy, we report the results in terms of five evaluation metrics with two of which also consider the ranking qualities [22]. The five evaluation metrics are: Precision@N, Recall@N, Mean Average Precision (MAP), Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain (NDCG).

In most cases, users only care about the topmost recommended items, we employ these evaluations at a given cut-off n. The definition are as follows.

$$Precision@n = \frac{\text{\# of items the user interacted in top n}}{n} \tag{7}$$

$$Recall@n = \frac{\text{\# of items the user interacted in top n}}{\text{total \# of items the user interacted}} \tag{8}$$

The former two evaluation metrics ignore the ranked position. MAP is used to assess the average accuracy of the overall ranking lists. It is the mean of all average precisions (AP) over all relevant users. $\mathbf{1}_{rel}(i)$ is an indicator which equals to 1 if user u has interacted with item i, and 0, otherwise.

$$AP(u) = \frac{\sum_{j=1}^{N} precision@j \times \mathbf{1}_{rel}(j)}{\#\text{ of relevant items}} \tag{9}$$

In practice, to make the items that interest target users rank higher will enhance the quality of recommendation lists. Therefore, we also employ two popular rank-aware evaluation metrics: MRR and NDCG. MRR cares about the single highest-ranked relevant item and it calculates the reciprocal of the rank at which the first item was put. NDCG evaluates the ranking quality of the overall recommendation list. The definition of MRR and NDCG are as follows:

$$MRR = \frac{1}{M}\sum_{u=1}^{M}\frac{1}{rank_u} \tag{10}$$

Here, $rank_u$ is the rank of the first correct item for user u.

$$DCG_n = \sum_{i=1}^{n}\frac{\mathbf{1}_{rel}(i)}{log_2 i + 1} \tag{11}$$

And $NDCG_n = DCG_n/IDCG_n$ with $IDCG_n$ denoting the DCG for perfect ranked list.

4.3 Comparison Baselines

We compare our approach with the following seven traditional and recent advanced baselines:

- **Random.** We randomly select Meetup groups from all possible candidates and recommend them to users.
- **MostPopular.** It is a non-personalized method which generates recommendations based on item popularity and recommends users with the most popular items.
- **ItemKNN** [2]**.** Item-based collaborative filtering method recommends items which are similar to other items the user has liked. Here the similarity between items is computed with cosine function.
- **BPRMF** [21], BPRMF is a competitive baseline for ranking prediction. It also employs a pairwise ranking loss and is optimized with a Bayesian Personalized Ranking algorithm on implicit feedback.
- **WRMF** [12], This algorithm is specified for one-class recommendation. It minimizes the squared errors in a pointwise manner and adopts a weight strategy to control the gradients for each user and item latent factors.

- **SLIM** [20], SLIM is a top-n recommendation model. It uses a sparse linear method to generate recommendation by aggregating user purchase and rating profiles. We optimize the objective function in a Bayesian personalized ranking criterion due to efficiency consideration.
- **CML** [10], collaborative metric learning considers the distances between users and items, and adopts the metric learning idea to learn user and item vectors. Here, we train this model with hinge loss (without WARP) due to the scalability issue of WARP loss [27].

For Random, mostPOP, ITEMKNN, BPR-MF, WRMF, SLIM, we use the implementation of Mymedialite [3]. We implemented our approach and CML with Tensorflow[2]. Since SLIM and CML are proved to perform better than many baselines such as [25], we do not further report them.

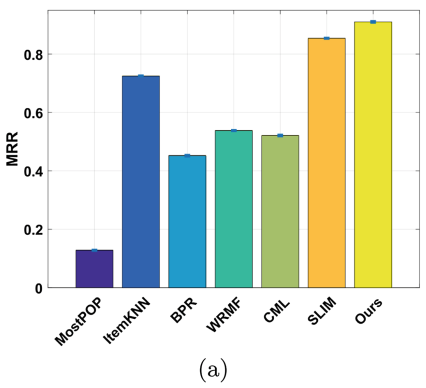
Table 3. Performance comparison in terms of precision@5, precision@10, recall@5, recall@10 and MAP on Meetup San Francisco.

Method	Precision@5	Precision@10	Recall@5	Recall@10	MAP
Random	0.002 ± 0.001	0.002 ± 0.002	0.001 ± 0.001	0.003 ± 0.001	0.004 ± 0.002
POP	0.041 ± 0.001	0.034 ± 0.001	0.030 ± 0.002	0.049 ± 0.001	0.039 ± 0.001
ItemKNN	0.362 ± 0.001	0.247 ± 0.002	0.312 ± 0.002	0.410 ± 0.003	0.346 ± 0.003
BPR	0.205 ± 0.002	0.162 ± 0.003	0.177 ± 0.002	0.275 ± 0.003	0.205 ± 0.004
WRMF	0.254 ± 0.001	0.194 ± 0.002	0.220 ± 0.001	0.323 ± 0.003	0.248 ± 0.002
CML	0.238 ± 0.002	0.188 ± 0.001	0.208 ± 0.003	0.321 ± 0.002	0.245 ± 0.003
SLIM	0.506 ± 0.002	0.340 ± 0.003	0.459 ± 0.002	0.579 ± 0.002	0.516 ± 0.001
Ours	**0.592 ± 0.002**	**0.392 ± 0.001**	**0.523 ± 0.002**	**0.642 ± 0.003**	**0.611 ± 0.001**

4.4 Implementation Details

We implement our model with Tensorflow and test it on a Linux machine. All learning parameters are initialized with random normal distribution and we use Adam algorithm [14] to learn the optimal parameters. Hyper-parameters are tuned based on grid search. For the deep neural components, we use two hidden layers with constant structure with 20 neurons for each layer. The output dimension and k are set to 10. We use *tanh* as the nonlinear activations. The inputs of the deep neural component are the categories of the meetup groups. The learning rate is set to 0.001 and the regularization rate λ is set to 0.001. Batch size is set to 1024. The scaling factor τ is set to 2. The dynamic negative sampling size t is set to 5. We randomly split each dataset into a training set and a testing set by the ratio of 5:1, and report the average results over five different splits. Parameters of other baselines are also tuned carefully to achieve the best performances.

[2] https://www.tensorflow.org/.



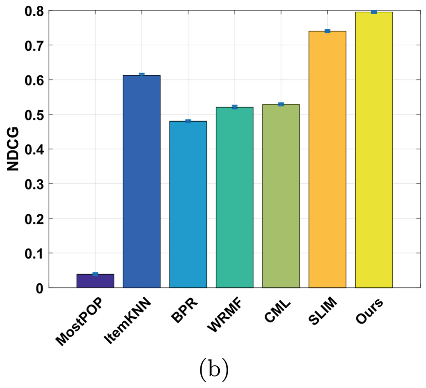


Fig. 4. MRR (a) and NDCG (b) comparison on Meetup San Francisco (We omit Random as it performs poorly).

4.5 Results and Analysis

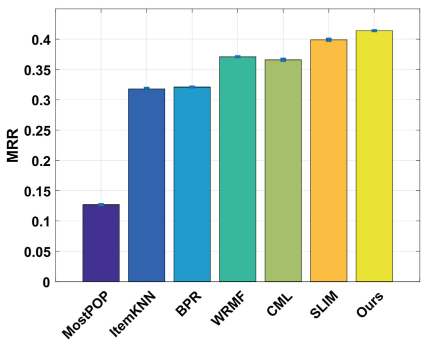
Tables 3, 4 and Figs. 4, 5 show the performance comparison on the two datasets. We observe that our model outperforms all other baselines in terms of both accuracy and ranking qualities. The overall improvements on Meetup San Francisco is about 12.53%, and that on Meetup New York city is about 5.08%. We find that latent factor models such as BPR and WRMF do not work well on both datasets, especially on Meetup San Francisco, which might be caused by the extreme sparsity. The performances of CML are slightly worse than WRMF. Similarity based model ItemKNN works well on Meetup New York city but it is computational expensive at prediction stage. SLIM is a very strong baseline. Our model is built upon SLIM, but our model outperforms SLIM by a large margin. The main reason is that our model can capture the content of the items and optimize the results with a more reasonable sampling strategy.

In addition, we also compare our model with SLIM in terms of convergence speed. Figure 6 shows the varying MAP, NDCG, Precision@10 and Recall@10

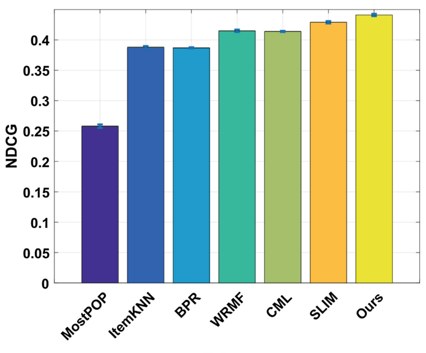
Table 4. Performance comparison in terms of precision@5, precision@10, recall@5, recall@10 and MAP on Meetup New York City.

Method	Precision@5	Precision@10	Recall@5	Recall@10	MAP
Random	0.001 ± 0.001	0.001 ± 0.002	0.002 ± 0.001	0.001 ± 0.001	0.003 ± 0.002
POP	0.041 ± 0.001	0.034 ± 0.001	0.036 ± 0.000	0.058 ± 0.001	0.040 ± 0.001
ItemKNN	0.134 ± 0.001	0.106 ± 0.001	0.124 ± 0.000	0.194 ± 0.001	0.132 ± 0.001
BPR	0.129 ± 0.000	0.102 ± 0.001	0.119 ± 0.001	0.186 ± 0.003	0.129 ± 0.002
WRMF	0.157 ± 0.000	0.122 ± 0.001	0.147 ± 0.002	0.223 ± 0.001	0.156 ± 0.001
CML	0.154 ± 0.002	0.120 ± 0.001	0.143 ± 0.001	0.219 ± 0.003	0.153 ± 0.001
SLIM	0.168 ± 0.001	0.128 ± 0.003	0.159 ± 0.003	0.238 ± 0.002	0.167 ± 0.001
Ours	**0.178 ± 0.002**	**0.136 ± 0.001**	**0.168 ± 0.001**	**0.249 ± 0.002**	**0.178 ± 0.002**

of our model, SLIM and ItemKNN on dataset Meetup San Francisco with the increase of training epochs. We find that our model converges much faster than SLIM, and it only takes about 15 iterations to achieve the best performance. This is mainly due to the dynamic negative sampling method we adopted as this sampling strategy can help our model to find comparably informative negative samples.



(a)



(b)

Fig. 5. MRR (a) and NDCG (b) comparison on Meetup New York City.

5 Related Work

In this section, we briefly review the related work of event recommendation and deep learning based recommendation.

5.1 Deep Learning for Recommender System

In recent years, deep learning has been revolutionizing the recommender systems. The achievements of deep learning based recommender systems in both industry and academia are inspiring and enlightening [28]. There are a various of deep learning techniques [5], and most of them can be applied to recommendation tasks somehow. For example, Convolutional Neural Network (CNN) can be used to extract features from textual [13] and visual information [6] of items and users. Recurrent Neural Network (RNN) is capable of modeling the temporal dynamics and sequential patterns of historical interactions [9]. Autoencoder can learn salient feature representations from side information to enhance recommendation quality [25,29,30]. We can even combine several deep learning technologies together to form a powerful composite recommendation model. Deep learning algorithms can also be integrated into conventional recommendation methods such as matrix factorization, factorization machine and collaborative metric learning [7,8,23]. There are two major motivations in applying deep learning techniques to recommender systems. First, Deep learning is powerful

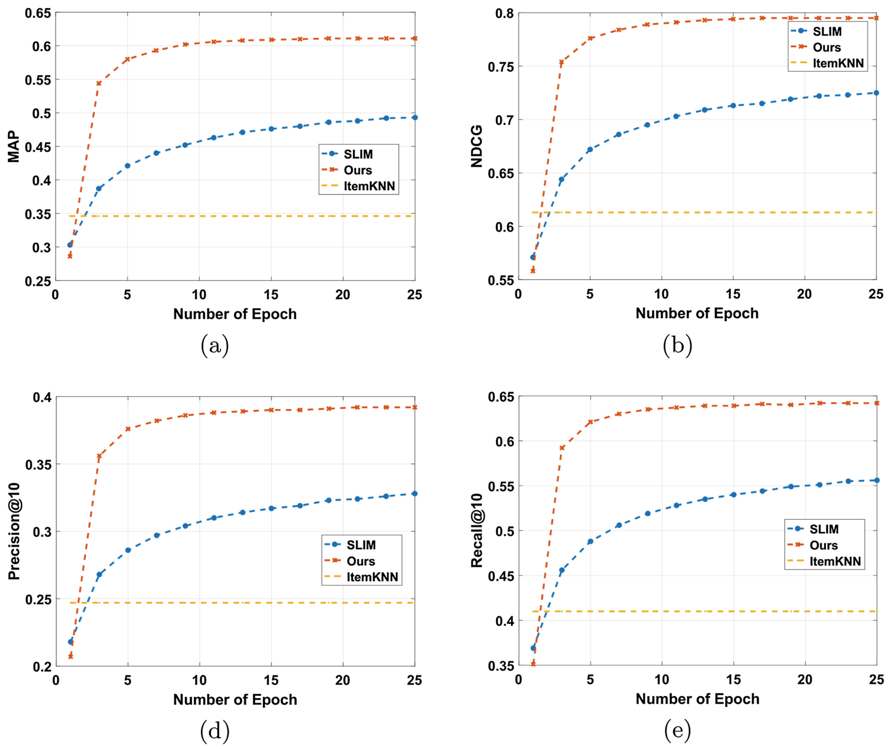


Fig. 6. Convergence of our model and SLIM in terms of (a) MAP, (b) NDCG, (c) Precision@10, (d) Recall@10. Overall, our model converges much faster than SLIM.

in representation learning [1], thus it also provides a desirable tool for feature learning in recommender systems [24]. Second, with nonlinear activations, we can add nonlinearity to recommendation models to capture intricate and complex characteristics of real-world datasets.

5.2 Event Recommendation

Another related work is about event recommendation since Meetup meeting is also a kind of event. Note that, in this work, we mainly focus on recommending Meetup groups for users to join in rather than recommending Meetup meetings (The organizer of Meetup groups can host Meetup meetings regularly, so Meetup groups recommendation and Meetup meetings recommendation are two different tasks for Meetup service recommendation.), nonetheless, it is an important task that we want to solve in the future. [17] proposed an event recommendation methodology based on graph random walking and history preference reranking. They obtain the candidate events by executing random walking on a hybrid graph consisting of different types of nodes to represent available entities in an event-based social network. Then they extract user preferences from her attended events and compute the similarities between her interests and her candidate events. Finally, recommended event lists are obtained by combining the two similarity scores. [26] proposed a Social Information Augmented Recommender System (SIARS), which fully exploits the social influence of event hosts and group members together with basic context information for event recommendation. [16] formulated multiple interactions among users, events, groups and locations into an unified framework and proposed a collective pairwise matrix factorization (CPMF) model to estimate users' pairwise preferences on events, groups and locations. [18] proposed a successive event recommender system based on graph entropy (SERGE) to deal with the new event cold start problem by exploiting diverse relations as well as asynchronous feedback in EBSNs. [19] proposed a new link prediction method for the Meetup social network, which recommends events to users according to the events they participated in and their field of interests. [4] proposed a Bayesian latent factor model (denoted as SogBmf) for event recommendation, based on the matrix factorization framework, to integrate social group influence with individual preference.

6 Conclusion and Future Work

In this paper, we proposed a coupled linear and deep nonlinear model for Meetup service recommendations. Our model can not only model the historical interaction patterns but also learn the item features effectively. We explored a novel logarithm loss for pairwise training of the proposed model. To further enhance the accuracy, we adopted a dynamic negative sampling strategy to select informative negative samples, which can improve the performance and lead to faster convergences. Experiments on two real-world large-scale Meetup datasets showed that our model can achieve the best performances for Meetup service recommendations.

In the further, we will explore integrate contextual information such as date, location, social network and weather to better anticipate user's intentions so as to make more satisfying recommendations. We will also explore methods for better Meetup meetings recommendation to enhance the Meetup service user experience.

References

1. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013)
2. Deshpande, M., Karypis, G.: Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst. (TOIS) **22**(1), 143–177 (2004)
3. Gantner, Z., Rendle, S., Freudenthaler, C., Schmidt-Thieme, L.: Mymedialite: a free recommender system library. In: Proceedings of the Fifth ACM Conference on Recommender Systems, pp. 305–308. ACM (2011)
4. Gao, L., Wu, J., Qiao, Z., Zhou, C., Yang, H., Hu, Y.: Collaborative social group influence for event recommendation. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 1941–1944. ACM (2016)

5. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep Learning, vol. 1. MIT Press, Cambridge (2016)
6. He, R., McAuley, J.: VBPR: visual Bayesian personalized ranking from implicit feedback. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI 2016, pp. 144–150. AAAI Press (2016). http://dl.acm.org/citation.cfm?id=3015812.3015834
7. He, X., Chua, T.S.: Neural factorization machines for sparse predictive analytics. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 355–364. ACM (2017)
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, pp. 173–182. International World Wide Web Conferences Steering Committee (2017)
9. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939 (2015)
10. 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, pp. 193–201. International World Wide Web Conferences Steering Committee (2017)
11. Hsieh, C.K., Yang, L., Wei, H., Naaman, M., Estrin, D.: Immersive recommendation: news and event recommendations using personal digital traces. In: Proceedings of the 25th International Conference on World Wide Web, WWW 2016, pp. 51–62. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2016). https://doi.org/10.1145/2872427.2883006
12. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: Eighth IEEE International Conference on Data Mining, ICDM 2008, pp. 263–272. IEEE (2008)
13. Kim, D., Park, C., Oh, J., Lee, S., Yu, H.: Convolutional matrix factorization for document context-aware recommendation. In: Proceedings of the 10th ACM Conference on Recommender Systems, pp. 233–240. ACM (2016)
14. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
15. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Comput. **7**(1), 76–80 (2003)
16. Liu, C.Y., Zhou, C., Wu, J., Xie, H., Hu, Y., Guo, L.: CPMF: a collective pairwise matrix factorization model for upcoming event recommendation. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 1532–1539. IEEE (2017)
17. Liu, S., Wang, B., Xu, M.: Event recommendation based on graph random walking and history preference reranking. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 861–864. ACM (2017)
18. Liu, S., Wang, B., Xu, M.: Serge: successive event recommendation based on graph entropy for event-based social networks. IEEE Access (2017)
19. Müngen, A.A., Kaya, M.: A novel method for event recommendation in meetup. In: Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 959–965. ACM (2017)
20. Ning, X., Karypis, G.: Slim: sparse linear methods for top-n recommender systems. In: 2011 IEEE 11th International Conference on Data Mining, pp. 497–506, December 2011

21. 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, UAI 2009, pp. 452–461. AUAI Press, Arlington, Virginia, United States (2009). http://dl.acm.org/citation.cfm?id=1795114.1795167
22. Shani, G., Gunawardana, A.: Evaluating recommendation systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P. (eds.) Recommender Systems Handbook, pp. 257–297. Springer, Boston (2011). https://doi.org/10.1007/978-0-387-85820-3_8
23. Tay, Y., Anh Tuan, L., Hui, S.C.: Latent relational metric learning via memory-based attention for collaborative ranking. In: Proceedings of the 2018 World Wide Web Conference, WWW 2018, pp. 729–739. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2018). https://doi.org/10.1145/3178876.3186154
24. Tay, Y., Tuan, L.A., et al.: Multi-pointer co-attention networks for recommendation. CoRR abs/1801.09251 (2018). http://arxiv.org/abs/1801.09251
25. Wang, H., Wang, N., Yeung, D.Y.: Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2015, pp. 1235–1244. ACM, New York (2015). https://doi.org/10.1145/2783258.2783273
26. Wang, Z., Zhang, Y., Li, Y., Wang, Q., Xia, F.: Exploiting social influence for context-aware event recommendation in event-based social networks. In: INFOCOM 2017-IEEE Conference on Computer Communications, pp. 1–9. IEEE (2017)
27. 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)
28. Zhang, S., Yao, L., Sun, A.: Deep learning based recommender system: a survey and new perspectives. arXiv preprint arXiv:1707.07435 (2017)
29. Zhang, S., Yao, L., Xu, X.: AutoSVD++: an efficient hybrid collaborative filtering model via contractive auto-encoders. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2017, pp. 957–960. ACM, New York (2017). https://doi.org/10.1145/3077136.3080689
30. Zhang, S., Yao, L., Xu, X., Wang, S., Zhu, L.: Hybrid collaborative recommendation via semi-autoencoder. In: Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, E.S. (eds.) ICONIP 2017. LNCS, vol. 10634, pp. 185–193. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70087-8_20
31. Zhang, W., Chen, T., Wang, J., Yu, Y.: Optimizing top-n collaborative filtering via dynamic negative item sampling. In: Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 785–788. ACM (2013)