Session-based Recommendation with Hierarchical Memory Networks

Bo Song
Zhejiang University
Hangzhou, China
bosong16@zju.edu.cn

Yi Cao
Zhejiang University
Hangzhou, China
cao_yi@zju.edu.cn

Weifeng Zhang
Zhejiang University
Hangzhou, China
zhangwf@zju.edu.cn

Congfu Xu*
Zhejiang University
Hangzhou, China
xucongfu@zju.edu.cn

ABSTRACT

The task of session-based recommendation aims to predict users' future interests based on anonymous historical sessions. Recent works have shown that memory models, which capture user preference from previous interaction sequence with long short-term or short-term memory, can lead to encouraging results in this problem. However, most existing memory models tend to regard each item as a memory unit, which neglect *n*-gram features and are insufficient to learn the user's feature-level preferences. In this paper, we aim to leverage *n*-gram features and model users' feature-level preferences in an explicit and effective manner. To this end, we present a memory model with multi-scale feature memory for session-based recommendation. A densely connected convolutional neural network (CNN) with short-cut path between upstream and downstream convolutional blocks is applied to build multi-scale features from item representations, and features in the same scale are combined with memory mechanism to capture users' feature-level preferences. Furthermore, attention is used to adaptively select users' multi-scale feature-level preferences for recommendation. Extensive experiments conducted on two benchmark datasets demonstrate the effectiveness of the proposed model in comparison with competitive baselines.

KEYWORDS

Session-based recommendation; Memory networks; Densely connected CNN; Attention mechanism;

ACM Reference Format:
Bo Song, Yi Cao, Weifeng Zhang, and Congfu Xu. 2019. Session-based Recommendation with Hierarchical Memory Networks. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM'19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3357384.3358120

*Corresponding author.

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358120

1 INTRODUCTION

Recommender systems are fundamental components of modern web services. A major application scenario of recommender systems is session-based recommendation, where the user identification is unknown and only the historical clicks during an ongoing session are available. In such scenario, predicting users' actions is quite challenging. Recently, memory mechanism [1, 2, 8] has been introduced to model the user's sequential behavior and has achieved promising results. For example, STAMP [8] designs a short-term memory priority model to capture the user's general interests and current interests simultaneously for session-based recommendation.

However, most of the memory-augmented recommendation models only store users' historical actions in a primitive way: they tend to regard each item as a memory unit and only capture the user's item-level preference, without taking *n*-gram features and user's feature-level preferences into account. In fact, several continuous clicked items can be used to extract common features, just like continuous words that form a phrase in a sentence can be used to extract *n*-gram feature. For example, if a user has clicked an iphone and a macbook, it is highly likely that the user is interested in the common feature of the two products—Apple's products. We argue that users' feature-level preferences can be leveraged to further improve the predictive accuracy of recommendation models.

In this paper, we propose a session-based recommendation model with Hierarchical Memory Networks (HMN). Our model mainly consists of two components, i.e., a multi-scale feature generation component and a hierarchical memory network component. The first one is a densely connected CNN model inspired by [4, 13]. The basic idea is to use CNN to extract *n*-gram features from a session sequence. However, directly utilizing convolutional filters with different window sizes to extract such multi-scale features fails to take interactions of feature maps from different filter sizes into account. Thereby a densely connected CNN which uses dense connections between different convolutional blocks is adopted. Once the feature generation step is completed, we can obtain several external memory matrices which store the user's item-level or feature-level preferences. The second component, the hierarchical memory network component, utilizes neural attention mechanism to form a collective feature summary by adaptively placing weights on these

memory matrices. The attention mechanism is utilized in a hierarchical style: it first summarizes features of the same scale (in the same memory matrix), and then summarizes those attentive features of different scale (across memory matrices).

Our primary contributions can be summarized as follows:

- We propose a novel Hierarchical Memory Network (HMN) model for session-based recommendation. With the densely connected CNN component, our model is able to generate memory matrices which store users' item-level and feature-level preferences.
- Unlike existing memory models which construct session representation from item embeddings directly, our model constructs session representation with a hierarchical attention mechanism, which is able to adaptively select multi-scale features from the memory matrices for recommendation.
- Comprehensive experiments conducted on real-world datasets show that HMN consistently outperforms the state-of-the-art methods.

2 RELATED WORK

2.1 Session-based Recommendation

In the scenario of session-based recommendation, user profile is unavailable and only users' sequential actions are provided. Hidasi et al. [3] introduce RNN to model transitions between items in the sessions, which is latter improved by Tan et al. [12] with data augmentation. These works only consider users' sequential behavior in the current session, Li et al. [7] further propose a hybrid encoder with an attention mechanism (NARM) to account for users' main purpose. More recently, Liu et al. [8] propose a short-term memory priority model (STAMP) to capture users' general interests and explicitly take the effects of usersâĂŹ current actions on their next moves into account. Song et al. [11] further introduce social relations into session-based recommendation.

2.2 Memory Augmented Neural Networks

Typical memory augmented neural networks generally consist of two components: an external memory (e.g., a matrix) and a controller (e.g., a neural network) that performs operations on the memory. The memory matrix is used to maintain historical information, which is beneficial to increase model capacity and track long-term dependencies. The controller manipulates the maintained memories with reading or writing operations. Memory networks have been successfully adapted to recommendation [1, 2, 5]. Ebesu et al. [2] propose a neural architecture called Collaborative Memory Networks (CMN) to combine the power of latent factor models and neighborhood-based approaches. Chen et al. [1] also use a memory-augmented neural network to enhance the expressiveness of recommender. Recently, Huang et al. [5] incorporate knowledge base information with a Key-Value Memory Network (KV-MN) into a sequential recommender.

3 METHOD

Given the item set $\mathcal{I} = \{i_1, i_2, \cdots, i_{|\mathcal{I}|}\}$, and a session $s = [x_1, x_2, \cdots, x_{|s|}]$ with $x_i \in \mathcal{I}$, the goal of session-based recommendation is to predict the next item that the user will interact with. Figure 1 illustrates the framework of the proposed HMN model. Our model consists of two main components, i.e., a multi-scale feature generation component and a hierarchical memory network component, which will be presented in detail in the following subsections.

3.1 Multi-scale Feature Generation

The multi-scale generation module takes a session sequence s as input, and generates multi-scale features by a densely connected CNN. Densely connected CNN stacks multiple convolutional blocks, where each convolutional block is used to extract features from all the downstream blocks. In this way, the upstream convolutional blocks construct features for small n-grams, and downstream ones for large n-gram features. To demonstrate the use of densely connected CNN in this module, we introduce necessary preliminaries, i.e., convolutional block and dense connections first.

Convolutional Block. Given a session $s = [x_1, x_2, \cdots, x_{|s|}]$, each item x_i in the session is firstly transformed to a dense vector $\boldsymbol{e}_i \in \mathbb{R}^d$ by an embedding layer. Then the whole session can be represented as the concatenation of these dense vectors:

$$\mathbf{E} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_{|s|}]^T \in \mathbb{R}^{|s| \times d}. \tag{1}$$

A convolutional block consists of k convolutional filters, each of which is an $h \times d$ matrix. Performing convolution operation on the session representation $\mathbf{E}$ with the j-th convolutional filter $\mathbf{W}^j \in \mathbb{R}^{h \times d}$ can produce a feature map

$$\boldsymbol{c}^j = [c_1^j, c_2^j, \cdots, c_{|s|-h+1}^j] \in \mathbb{R}^{|s|-h+1}, 1 \le j \le k, \tag{2}$$

with

$$c_i^j = f(\mathbf{W}^j, \mathbf{E}_{i:i+h-1}) = a(\langle \mathbf{W}^j, \mathbf{E}_{i:i+h-1} \rangle_F + b), \tag{3}$$

where a is a non-linear activation function, $b \in \mathbb{R}$ is a bias term, $\mathbf{E}_{i:i+h-1} = [\boldsymbol{e}_i, \boldsymbol{e}_{i+1}, \cdots, \boldsymbol{e}_{i+h-1}]^T \in \mathbb{R}^{h \times d}$, and $\langle \cdot, \cdot \rangle_F$ is Frobenius inner product. Note that zero padding can be used to ensure that the output feature map has the same size as the length of the input session. In the rest of this paper, we assume zero padding is used and $\boldsymbol{c}^j \in \mathbb{R}^{|s|}$.

For all the k filters, the output of the convolutional block is a feature map matrix

$$\mathbf{M} = Conv(\mathcal{W}, \mathbf{E}) = [\boldsymbol{c}^1, \boldsymbol{c}^2, \cdots, \boldsymbol{c}^k] \in \mathbb{R}^{|s| \times k}, \tag{4}$$

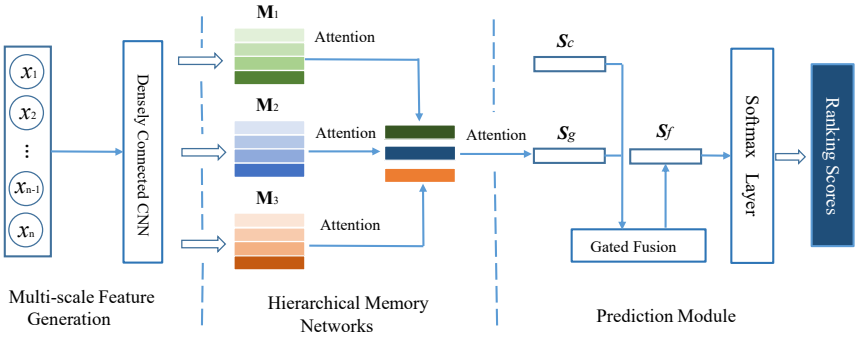
where $\mathcal{W} \in \mathbb{R}^{h \times d \times k}$ refers to all the k convolutional filters.

Dense Connections. Inspired by the work of [13] which adopts a densely connected CNN for text classification, we use dense connections between convolutional blocks to endow the feature generation module with the ability to compose representations from variable n-gram features. For the l-th layer of a densely connected CNN, the input is concatenation of the outputs of all upstream layers $[\mathbf{M}_1, \mathbf{M}_2, \cdots, \mathbf{M}_{l-1}] \in \mathbb{R}^{|s| \times k \times (l-1)}$, and the output $\mathbf{M}_l$ can be formulated as follows

$$\mathbf{M}_l = Conv(\mathcal{W}_l, [\mathbf{M}_1, \mathbf{M}_2, \cdots, \mathbf{M}_{l-1}]), \tag{5}$$

where $\mathcal{W}_l \in \mathbb{R}^{h \times k \times (l-1) \times k}$ refers to the corresponding k convolutional filters.

With the illustrations above, we now elaborate how the dense connections are used in our feature generation module. Figure 2 shows an example of a 3-layer feature generation module. At layer 1, the module uses k convolutional filter with filter size $1 \times d$ to



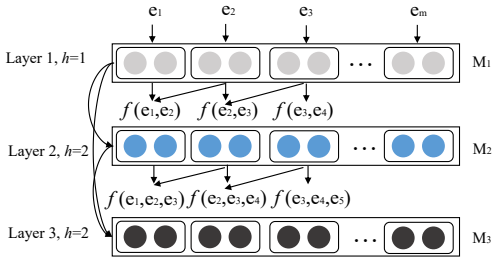


Figure 2: The convolution operations and dense connections in the feature generation module. The convolutional filters in function f are omitted.

convert the dimension of item embeddings from d to k to produce $\mathbf{M}_1$, and then generates $\mathbf{M}_2$ by applying k filters of size $2 \times k$ on $\mathbf{M}_1$. Similarly, $\mathbf{M}_3$ is generated by applying filters of size $2 \times k \times 2$ on the concatenation of $\mathbf{M}_1$ and $\mathbf{M}_2$. As a result, we obtain three feature matrices corresponding to unigram, bigram and trigram features, respectively.

3.2 Hierarchical Memory Networks

After feature generation module, we obtain multi-scale features which can be regarded as the user's item-level and feature-level preferences. These features encodes the user's previous records, and thereby we refer to them as the user's memory component. In order to effectively take advantage of these features, we present a hierarchical attention mechanism to adaptively select information from them for recommendation. Thereby we refer to this component as hierarchical memory networks.

As shown in figure 1, the hierarchical memory network first attentively pools features of the same scale, and it furthermore uses another attention net to summarize these attentive features to express the user's general interests. Formally, features of the same scale are pooled as follows

$$\boldsymbol{m}_l^a = \sum_{i=1}^{|s|} \alpha_l^i \boldsymbol{m}_l^i, \tag{6}$$

where $\boldsymbol{m}_l^i \in \mathbb{R}^k$ denotes the i-th row of the l-th feature map matrix $\mathbf{M}_l$, α_l^i is the attention score calculated by a feed-forward neural network (FNN). The FNN is defined as

$$\alpha_l^i = \boldsymbol{v}_l^T \sigma(\mathbf{V}_l^1 \boldsymbol{m}_l^i + \mathbf{V}_l^2 \boldsymbol{e}_l + \boldsymbol{b}_l), \tag{7}$$

where $\boldsymbol{v}_l \in \mathbb{R}^k$ is a weighting vector, $\mathbf{V}_l^1, \mathbf{V}_l^2 \in \mathbb{R}^{k \times k}$ are weighting matrices, $\boldsymbol{b}_l \in \mathbb{R}^k$ is a bias term, and $\boldsymbol{e}_l$ is a projection vector of the last item in the current session defined as

$$\overrightarrow{\boldsymbol{e}_l} = \mathbf{V}_l^3 \boldsymbol{e}_{|s|}, \tag{8}$$

with $\mathbf{V}_l^3 \in \mathbb{R}^{k \times d}$. Note that in Eq.(7), $\boldsymbol{e}_l$ is used to take the user's short-term memory into account, as suggested in [8]. After obtaining the attentive vectors $\boldsymbol{m}_l^a$ $(1 \le l \le L)$, another FNN is utilized to generate attention weights for each of these vectors to express the user's general interests in the current session. The FNN used for attention calculation is defined as

$$\beta_l = \boldsymbol{u}^T \sigma(\mathbf{U}_1 \boldsymbol{m}_l^a + \mathbf{U}_2 \boldsymbol{m}_s + \boldsymbol{b}), \tag{9}$$

where $\boldsymbol{m}_s$ is defined as

$$\boldsymbol{m}_s = \frac{1}{L} \sum_{l=1}^{L} \boldsymbol{m}_l^a. \tag{10}$$

Finally, the user's interests in general $\boldsymbol{s}_g$ with respect to the current session can be computed as follows

$$\boldsymbol{s}_g = \sum_{l=1}^{L} \beta_l \boldsymbol{m}_l^a. \tag{11}$$

3.3 Prediction Module

Except for the general interests, we also use an embedding vector $\boldsymbol{s}_c$ to capture the user's current interests, which is simply defined as the embedding of the last-clicked item $\boldsymbol{e}_m$, i.e., $\boldsymbol{s}_c = \boldsymbol{e}_m$. We propose a gated fusion method to fuse the general interests and current interests of a user. Formally, the fusion function is given as follows

$$\begin{aligned} \boldsymbol{\delta} &= \sigma(\mathbf{W}_g \boldsymbol{s}_g + \mathbf{W}_c \boldsymbol{s}_c + \boldsymbol{b}_g), \\ \boldsymbol{s}_f &= (1 - \boldsymbol{\delta}) \odot \boldsymbol{s}_g + \boldsymbol{\delta} \odot \boldsymbol{s}_c, \end{aligned} \tag{12}$$

where $\boldsymbol{s}_f$ is the final embedding vector of the session.

The prediction score of each item i is the inner product between $\boldsymbol{e}_i$ and $\boldsymbol{s}_f$

$$\hat{z}_i = \boldsymbol{s}_f^T \boldsymbol{e}_i. \tag{13}$$

The output scores $\hat{\boldsymbol{y}}$ is obtained by applying a softmax function to $\hat{\boldsymbol{z}} = [\hat{z}_1, \hat{z}_2, \cdots, \hat{z}_{|I|}]$

$$\hat{\boldsymbol{y}} = softmax(\hat{\boldsymbol{z}}). \tag{14}$$

The loss function is the cross-entropy loss between $\hat{\boldsymbol{y}}$ and $\boldsymbol{y}$.

4 EXPERIMENTS

4.1 Datasets

Two standard transaction datasets are used to evaluate the performance of different methods, i.e., YOOCHOOSE dataset and DIGINETICA dataset. Following previous work [7, 8], for both datasets, we filter out all sessions of length 1 and items appearing less than 5 times. Because the YOOCHOOSE dataset is quite large, we only use the recent fractions 1/64 of training sessions. For YOOCHOOSE, the sessions of subsequent day are used for test; while for DIGINETICA, the sessions of subsequent week are used for test.

4.2 Baselines

To evaluate the performance of the proposed method, we compare it with eight baselines, including four traditional methods (i.e., **PopRank**, **S-POP**, **BPR-MF** [9], **FPMC** [10]), two RNN-based methods (i.e.,**GRU4REC** [3], **GRU4REC+** [12]) and two attention/ memory-based methods (i.e., **NARM** [7], **STAMP** [8]).

All methods are evaluated in terms of the following two metrics: **R@**K (Short for Recall@K) and **MRR@**K.

4.3 Experimental Setup

We use ADAM [6] optimizer with a mini-batch of 512 and a learning rate of 0.001. For convenience, the maximum length of a sequence is set to 19 as the setting in NARM. The embedding dimension of the proposed HMN model is set to 30, and the number of convolutional blocks in densely connected CNN is set to 4.

Table 1: Performance comparison of HMN with baseline methods on the two datasets.

Datasets	YOOCHOOSE		DIGINETCA	
Metrics	R@20	MRR@20	R@20	MRR@20
PopRank	6.71	1.65	0.91	0.23
S-POP	30.44	18.35	21.07	14.69
BPR-MF	31.31	12.08	15.19	8.63
FPMC	45.62	15.01	31.55	8.92
GRU4REC	60.64	22.89	43.82	15.46
GRU4REC+	67.84	29.00	57.95	24.93
NARM	68.32	28.76	62.58	27.35
STAMP	68.74	29.67	62.03	27.38
HMN	**69.46**	**30.27**	**63.29**	**30.04**

4.4 Result Analysis

The results of all recommendation models on the two datasets are summarized in Table 2, where the best performance of each column is highlighted in boldface. As can be seen, HMN achieves the best performance in terms of R@20 and MRR@20 on both datasets, which illustrates the effectiveness of the proposed model. The recommendation quality of PopRank is very low, while S-POP can achieve comparable performance with personalized models (e.g., BPR-MF and FPMC) , which indicates that session context is very important when making recommendation. Personalized models only perform slight better than S-POP, which verifies that personalized models designed for personalized tasks are not good choices for session-based recommendation. In general, the performances of traditional models (e.g., PopRank, S-POP, BPR-MF and FPMC) are not competitive, and neural network baselines significantly outperform conventional models. NARM and STAMP are two strong baselines, both of them outperforms GRU4REC by a considerable margin. Nevertheless, our HMN model achieves the best results on both datasets in terms of both metrics. Compared with NARM and STAMP, HMN further takes advantage of n-gram features and achieves 1.04% and 2.02% improvements on R@20, 2.03% and 9.72% on MRR@20 on the two datasets respectively. The results illustrates that session representation generated from multi-scale features is more effective than from raw item embeddings.

5 CONCLUSION

In this paper, we propose a hierarchical memory networks for session-based recommendation. Empirical studies show that our model is able to outperform the state-of-the-arts on two benchmark datasets. As for future work, we will explore to incorporate side information into our model to further improve performance.

6 ACKNOWLEDGMENTS

We thank the support of National Natural Science Foundation of China No. 61672449.

REFERENCES

[1] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential Recommendation with User Memory Networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 108–116.

[2] Travis Ebesu, Bin Shen, and Yi Fang. 2018. Collaborative Memory Network for Recommendation Systems. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 515–524.

[3] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 241–248.

[4] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2261–2269.

[5] Jin Huang, Wayne Xin Zhao, Hong-Jian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *Proceedings of the 41st ACM International Conference on Research and Development in Information Retrieval, SIGIR*. 505–514.

[6] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[7] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1419–1428.

[8] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1831–1839.

[9] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 452–461.

[10] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*. ACM, 811–820.

[11] Weiping Song, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. 2019. Session-Based Social Recommendation via Dynamic Graph Attention Networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 555–563.

[12] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 17–22.

[13] Shiyao Wang, Minlie Huang, and Zhidong Deng. 2018. Densely Connected CNN with Multi-scale Feature Attention for Text Classification. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI*. 4468–4474.