BAR: Behavior-Aware Recommendation for Sequential Heterogeneous One-Class Collaborative Filtering

Mingkai He[1,2,3], Weike Pan[1,2,3*], Zhong Ming[1,2,3*]
mingkai_he@163.com, {panweike, mingz}@szu.edu.cn

[1]College of Computer Science and Software Engineering,
Shenzhen University, China

[2]National Engineering Laboratory for Big Data System Computing Technology,
Shenzhen University, China

[3]Guangdong Laboratory of Artficial Intelligence and Digital Economy (SZ),
Shenzhen, China

Motivation

The limitations of existing recommendation methods:

- Most sequential recommendation (a.k.a. SOCCF) methods cannot distinguish different types of behaviors for a same item in a sequence because they are designed for modeling one single type of behavior.
- Most existing heterogeneous sequential recommendation (a.k.a. SHOCCF) methods are based on RNN, which however needs to obtain the hidden state of each step in strict chronological order, resulting in low training efficiency.

Overall of Our Solution

- We propose a generic behavior-aware recommendation (BAR) framework, which is able to improve a typical sequential recommendation method such as
  GRU4Rec [Hidasi and Karatzoglou, 2018],
  SRGNN [Wu et al., 2019] and SASRec [Kang and McAuley, 2018].
- Specifically, we regard the representation module of an SOCCF method as a black box whose internal structure will not be changed. We improve its ability to distinguish different types of behaviors (e.g., examinations and purchases) from an input sequence of (item, behavior) pairs via our behavior attention layer, and obtain an improved representation of the sequence via our task-specific layer.

Advantages of Our Solution

- Our BAR can serve as a generic framework to adapt a typical sequential recommendation method from SOCCF to SHOCCF, which is suitable for most deep learning-based sequential recommendation methods that use the embedding matrix of an item sequence as input such as GRU4Rec and SASRec.

Related Work (1/3)

- **One-Class Collaborative Filtering (OCCF)**
  - FISM [Liang et al., 2018] replaces the user-specific feature vector in MF with the aggregated feature vectors of the items that a user has interacted, so that the final predicted score can be regarded as the similarity between the target item and the user's historical interacted items.
- **Heterogeneous One-Class Collaborative Filtering (HOCCF)**
  - RoToR [Pan et al., 2019] proposes a two-stage approach to use the examination data. In the first stage, a user's historical purchases and examinations are used in the item-oriented collaborative filtering (ICF) method to obtain a candidate list of items that the user will most likely purchase, which is then re-ranked by the model trained on the purchase data in the next stage.

Related Work (2/3)

- **Sequential One-Class Collaborative Filtering (SOCCF)**
  - SASRec [Kang and McAuley, 2018] is an attention-based model that extracts features from the input sequence through self-attention techniques.
- **Sequential Heterogeneous One-Class Collaborative Filtering (SHOCCF)**
  - BINN [Li et al., 2018] captures the representation of the current consumption motivations by a contextual LSTM (CLSTM), and learns a user's historical stable preferences with a bidirectional CLSTM (Bi-CLSTM).

Related Work (3/3)

Table: A brief summary of the four related problems w.r.t. the characteristics of feedback, i.e., homogeneous vs. heterogeneous, and sequential vs. non-sequential, including one-class collaborative filtering (OCCF), sequential OCCF (SOCCF), heterogeneous OCCF (HOCCF), and sequential HOCCF (SHOCCF).

	Homogeneous	Heterogeneous
Non-sequential	FISM [Kabbur et al., 2013], etc. (OCCF, many)	RoToR [Pan et al., 2019], etc. (HOCCF, some)
Sequential	SASRec [Kang and McAuley, 2018], etc. (SOCCF, many)	RIB [Zhou et al., 2018b], BINN [Li et al., 2018] **(SHOCCF, few)**

Problem Definition

Sequential Heterogeneous One-Class Collaborative Filtering (SHOCCF)

- Input: A historical heterogeneous sequence $\mathcal{S}_u = \{(i_u^{t-L+1}, f_u^{t-L+1}), ..., (i_u^{\ell}, f_u^{\ell}), ..., (i_u^{t}, f_u^{t})\}$, where (i_u^t, f_u^t) denotes the (item, behavior) pair at timestamp t w.r.t. user u.

- Goal: Predict the next likely-to-purchase item i of a user u from $\mathcal{I}$ at timestamp $t+1$.

Challenges

- Heterogeneity of users' behaviors in SHOCCF. For almost all the methods of SOCCF, they usually consider the item sequences with purchase behaviors and ignore the examination behaviors. However, in SHOCCF, there are a large number of examination behaviors that are mixed in the input sequence.
- Purchase-oriented item recommendation in SHOCCF. In SHOCCF, training with the next-item prediction loss cannot directly lead to a purchase-oriented recommendation model because the model's output cannot distinguish examinations and purchases well.
- Lacking a generic framework for adapting a sequential method from SOCCF to SHOCCF. Most SOCCF methods are based on different neural network architectures and they cannot be directly applied to SHOCCF.

BAR (1/3)

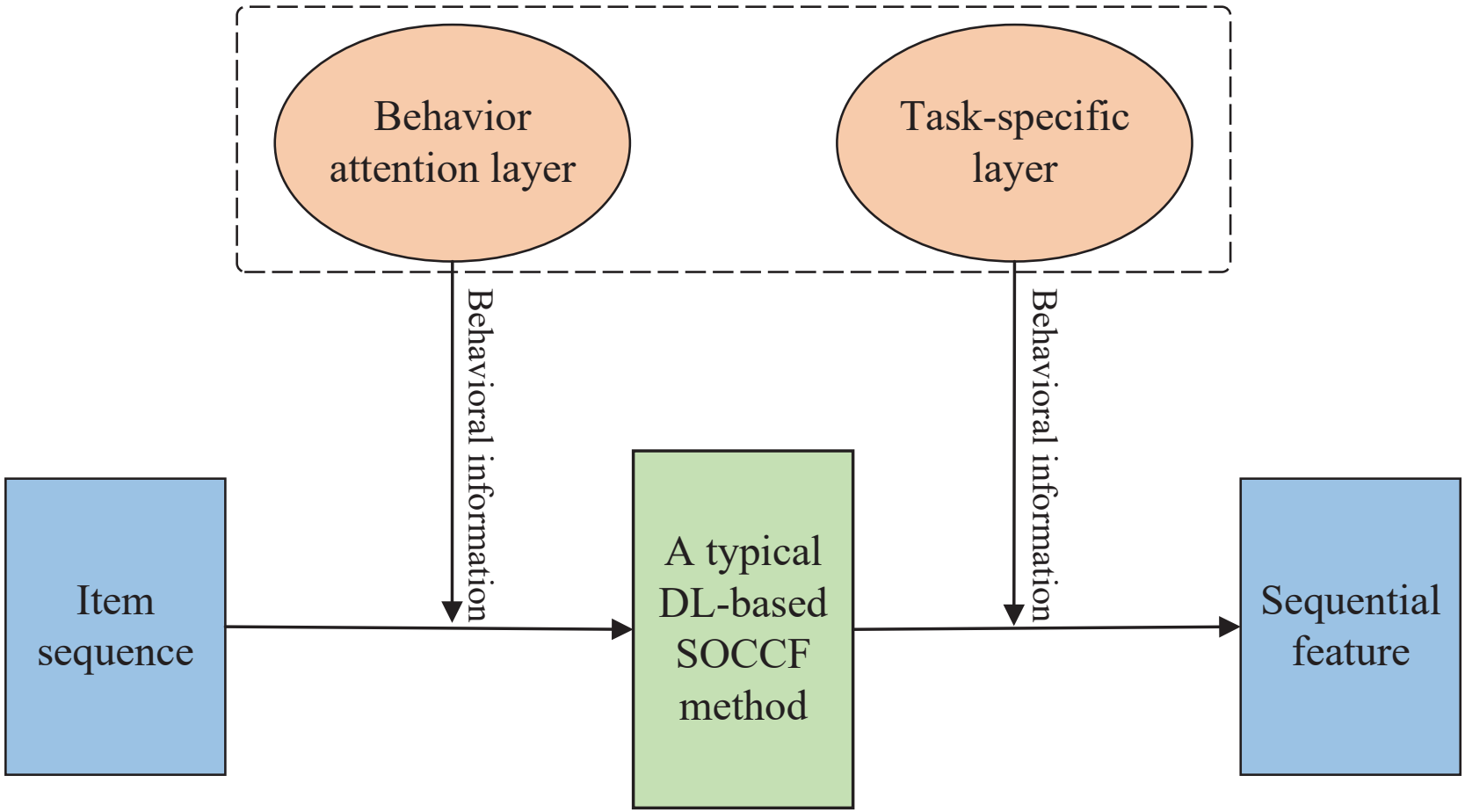


Figure: Overall solution of our behavior-aware recommendation (BAR).

BAR (2/3)

- As a response to the first challenge, we design a behavior attention layer to model the relationship between the target behavior and each behavior in the current sequence, and obtain the attention weight of each item. Notice that the attention weight will be fed into the input of the sequential recommendation method, so that the model can distinguish different behaviors in the historical item sequence.
- As a response to the second challenge, we design a task-specific layer that incorporates the user's real behavior at the next timestamp into the model as prior knowledge and enables the model to distinguish different tasks in the training phase.
- As a response to the last challenge, we can see that our solution is independent of the backbone recommendation method, which thus addresses the third challenge of developing a generic framework.

BAR (3/3)

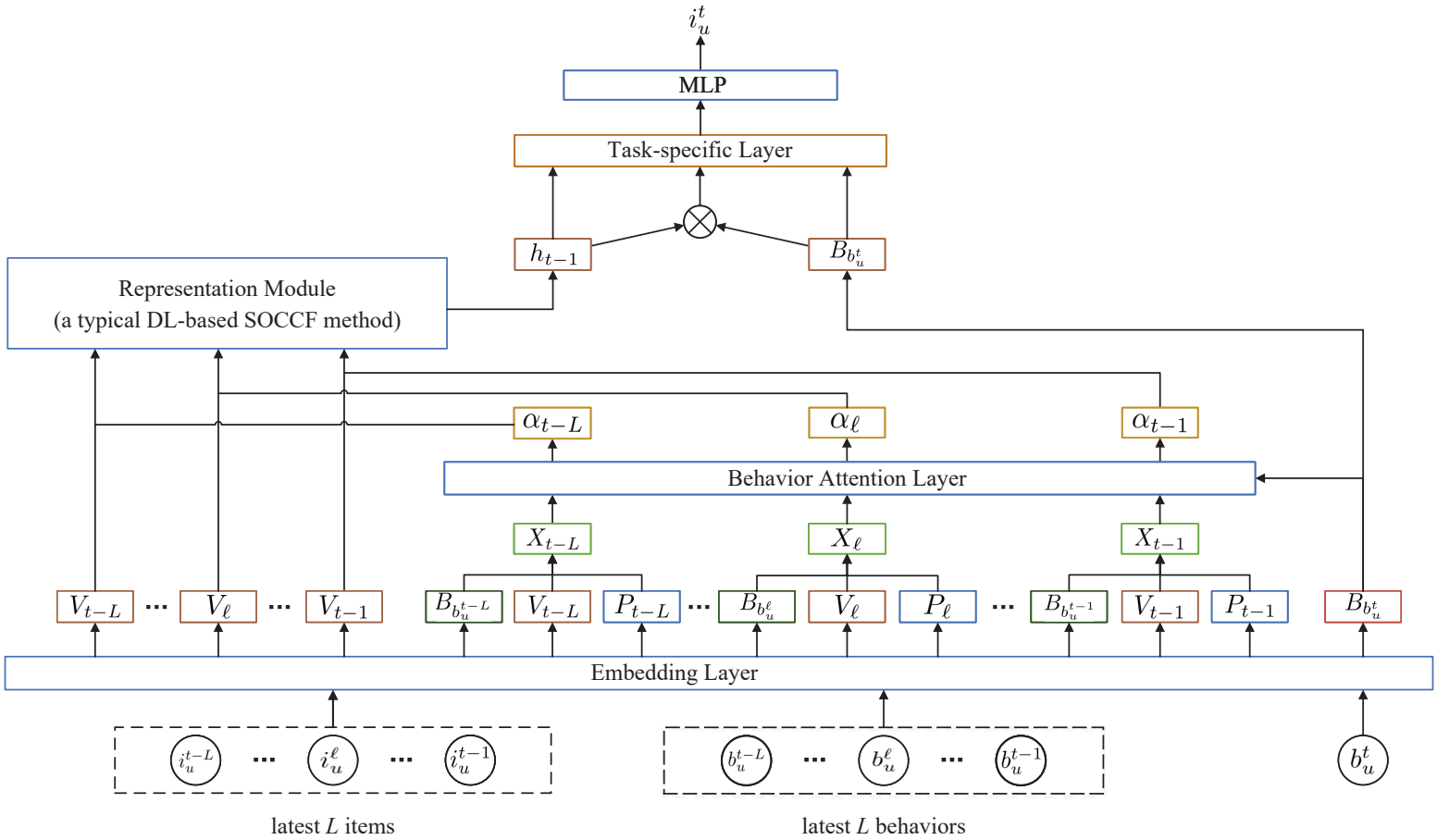


Figure: Illustration of our behavior-aware recommendation (BAR) for sequential heterogeneous one-class collaborative filtering (SHOCCF).

Representation Module (1/2)

- Most sequential recommendation methods with one single type of behavior (i.e., SOCCF) have their own ways to model a historical interaction sequence in order to get a feature representation of the sequence, which is usually called a representation module (RM),

$$h_{t-1} = RM(V_{i_u^{t-L}}, V_{i_u^{t-L+1}}, \ldots, V_{i_u^{t-1}}), \tag{1}$$

where $V_{i_u^{t-1}} \in \mathbb{R}^{d \times 1}$ denotes the item-specific latent vector of item i_u^{t-1} and $h_{t-1} \in \mathbb{R}^{d \times 1}$ is a sequential feature generated by RM.

Representation Module (2/2)

- RM is an important module for modeling the historical item sequences in various sequential recommendation methods such as the gated recurrent unit (GRU) in GRU4Rec and the CNN module in Caser.
- However, these methods focus on modeling sequential one-class feedback but cannot distinguish different types of behaviors, e.g., examinations and purchases. For this reason, Eq.(1) cannot be directly applied to SHOCCF.
- In our solution, we regard the representation module as a black box whose internal structure can be replaced by that of a certain SOCCF method but have a same format of input and output. We improve its ability to distinguish examinations and purchases from an input sequence of (item, behavior) pairs, and obtain an improved representation of the sequential feature.

Behavior Attention Layer (1/4)

- For each historical item sequence $\{i_u^{t-L}, \ldots, i_u^{t-1}\}$, the corresponding behavior sequence $\{b_u^{t-L}, \ldots, b_u^{t-1}\}$ is usually neglected in a typical SOCCF method, which is very important to well exploit them in SHOCCF.
- In addition, the position of a behavior in the sequence is also an important feature that affects the importance of the corresponding item.
- In the end, we also include the item information to form a vector X_ℓ that can represent the current position ℓ:

$$X_\ell = V_{i_u^\ell} + B_{b_u^\ell} + P_\ell, \tag{2}$$

where $V_{i_u^\ell} \in \mathbb{R}^{d\times 1}$, $B_{b_u^\ell} \in \mathbb{R}^{d\times 1}$ and $P_\ell \in \mathbb{R}^{d\times 1}$ are the item-specific latent vector of item i, behavior-specific latent vector of behavior b_u^ℓ and position-specific latent vector of position ℓ, respectively.

Behavior Attention Layer (2/4)

- When people purchase a certain product, their past historical behaviors will reflect different importance, so the model should pay more attention to a specific part of the items in the sequence rather than treating them equally.
- Inspired by the attention mechanism [Zhou et al., 2018a], we design a behavior attention layer and use it to calculate the relationship between the target behavior b_u^t and the current feature X_ℓ.
- We then obtain the input X_{att} of our behavior attention layer:

$$X_{att} = concat(X_\ell, X_\ell - B_{b_u^t}, X_\ell \odot B_{b_u^t}, B_{b_u^t}), \tag{3}$$

where $X_{att} \in \mathbb{R}^{4d \times 1}$ represents the information of the next behavior b_u^t and the current feature X_ℓ, and $\odot$ denotes the element-wise product.

Behavior Attention Layer (3/4)

- After that, we feed X_{att} to a two-layer MLP and get the attention weight α_ℓ:

$$H_\ell = ReLU(W_{att}X_{att} + b_{att}), \tag{4}$$

$$\alpha_\ell = Tanh(W_o H_\ell + b_o), \tag{5}$$

where $H_\ell \in \mathbb{R}^{d\times 1}$, $W_{att} \in \mathbb{R}^{d\times 4d}$, $b_{att} \in \mathbb{R}^{d\times 1}$, $\alpha_\ell \in \mathbb{R}$, $W_o \in \mathbb{R}^{1\times d}$ and $b_o \in \mathbb{R}$ are the parameters of the behavior attention layer. ReLU and Tanh are two commonly used nonlinear activation functions in deep learning, which enable the neural network to model nonlinear relationship among the features.

Behavior Attention Layer (4/4)

- In Eq.(1), we can see that different historical items are actually treated equally in the representation module. However, in a heterogeneous sequence, users may have different behaviors such as examining and purchasing the same item, and these behaviors reflect different degrees of importance the user attaches to the item.
- In response to this issue, our solution is to introduce the effect of different behaviors by increasing or decreasing the weight of the items:

$$h_{t-1} = RM((1+\alpha_{t-L})V_{i_u^{t-L}}, \ldots, (1+\alpha_{t-1})V_{i_u^{t-1}}), \tag{6}$$

where α_ℓ represents whether the model should pay more attention to the item i_u^ℓ with the behavior b_u^ℓ. For the coefficient $1+\alpha_\ell$ in Eq.(6), since we use the Tanh function in Eq.(5), we have $-1 < \alpha_\ell < 1$, and thus $0 < 1+\alpha_\ell < 2$.

Task-Specific Layer (1/4)

- When modeling a sequence with two different types of behaviors in SHOCCF, the task of the model is uncertain because the real behavior of the predicted next item may be purchase or examination, i.e., there are two different types of prediction tasks.
- Predicting the next interacted item in a sequence with both examinations and purchases actually includes two specific tasks, i.e., predicting the next examined item and predicting the next purchased item. Therefore, we model the difference between these two tasks through a task-specific layer.

Task-Specific Layer (2/4)

- Specifically, we use the actual behavior of the next timestamp, i.e., b_u^t, as prior knowledge to make the model learn the user's current intention, so that the model can understand the current prediction task in advance.
- We use an MLP to model the cross relationship between the sequential feature h_{t-1} and the behavior feature $B_{b_u^t}$. In this way, the sequential feature and the behavior feature are combined to obtain the cross feature z_t:

$$z_t = ReLU(W_z \times concat(h_{t-1}, B_{b_u^t}) + b_z), \tag{7}$$

where $W_z \in \mathbb{R}^{d \times 2d}$ and $b_z \in \mathbb{R}^{d \times 1}$ are learnable parameters, and $\times$ denotes matrix multiplication.

Task-Specific Layer (3/4)

- Then we obtain the input f_t of the final fully connected layer by concatenating h_{t-1}, z_t and $B_{b_u^t}$, which contains not only the sequential feature of the original model, but also the behavior feature that can distinguish the user's behavior at the next timestamp:

$$f_t = concat(h_{t-1}, z_t, B_{b_u^t}), \tag{8}$$

  where $f_t \in \mathbb{R}^{3d \times 1}$ includes the sequential feature h_{t-1} enhanced by our behavior attention layer, the real next behavior feature $B_{b_u^t}$ to help the model to distinguish these the two tasks, and their cross feature z_t.

- We feed f_t to a fully connected layer and obtain the score w.r.t. each item:

$$y = W_o f_t + b_o, \tag{9}$$

  where $W_o \in \mathbb{R}^{m \times 3d}$ and $b_o \in \mathbb{R}^{m \times 1}$ are learnable parameters.

Task-Specific Layer (4/4)

- At the end, the final score of each item is normalized by a softmax function to obtain the probability p_i of the item i being interacted at the next timestamp:

$$p_i = \frac{e^{y_i}}{\sum_{j=1}^{m} e^{y_j}}, \tag{10}$$

where y_i denotes the score of the item i generated by our solution. During the training phase, we use the following cross-entropy loss,

$$\mathcal{L} = -\sum_{i=1}^{m} Y_i \log(p_i) \tag{11}$$

where Y_i indicates the true label of item i, i.e., $Y_i = 1$ if a user interacts with item i at the next timestamp, and $Y_i = 0$ otherwise.

Datasets (1/4)

- We conduct experiments on the following four public datasets containing two different types of behaviors, i.e., examinations and purchases.
- MovieLens 1M (ML1M). It is a widely used movie rating dataset, containing about 1 million ratings for movies by 6,040 users, of which the rating values are from 1 to 5. We keep the (user, item) pairs with a rating value equal to 5 as the purchase behaviors, and the rest as the examination behaviors.
- Rec15. It is a dataset containing multiple anonymous sessions used in RecSys Challenge 2015, where the examinations and purchases in each session occur within a few minutes or hours. There are 9,249,729 sessions, 52,739 items, 33,000,944 examinations and 1,150,753 purchases.

Datasets (2/4)

- User Behavior (UB). It is a user behavior dataset collected from a real e-commerce website. It contains about 92 million examinations and purchases from 987,994 users and 4,162,024 items between November 25, 2017 and December 3, 2017.
- Tmall. Tmall is a dataset released at the IJCAI Competition 2015, which contains Taobao's shopping logs within 6 months. The dataset contains 424,170 users, 1,090,390 items, 48,550,713 examinations and 3,292,144 purchases. In addition, we remove the data on the day of Double Eleven (i.e., November 11) in order to avoid the interference caused by various promotion activities on that day.

Datasets (3/4)

We preprocess these datasets as follows:

- 1) we keep the records of examinations and purchases, and order the (item, behavior) pairs by the corresponding timestamps for each user;
- 2) we discard later duplicated (user, item, behavior) triples in a sequence in order to focus on recommending new items;
- 3) we discard cold-start items that are purchased fewer than 5 times for ML1M and Rec15, 10 for UB and 20 for Tmall;
- 4) we remove a sequence in which the number of purchases is smaller than 5 for ML1M, Rec15 and UB, and 10 for Tmall; and
- 5) for each interaction sequence, we take the last two purchases as the validation data and the test data, and the remaining as the training data.

Datasets (4/4)

Table: Statistics of the processed datasets used in the experiments.

Dataset	# Users	# Items	# Examinations	# Purchases	Avg. Length	Density
ML1M	5,645	2,357	628,892	223,305	150.96	6.41%
Rec15	36,917	9,621	446,442	233,263	18.41	0.45%
UB	20,858	30,793	470,731	136,250	29.10	0.09%
Tmall	17,209	16,176	831,117	240,901	62.29	0.38%

Evaluation Metrics

- We evaluate the top-k recommendation performance via two commonly used ranking-oriented metrics, i.e., hit ratio (HR@k) and normalized discounted cumulative gain (NDCG@k).
- HR@k represents the hit ratio of a user's target item in the recommended top-k items, which is used to measure the accuracy of the recommendation algorithm.
- NDCG@k is more concerned about the ranking position of the user's preferred target item in the top-k recommendation list.

Baselines (1/2)

- Five SOCCF algorithms:
  - GRU4Rec [Hidasi and Karatzoglou, 2018].
  - Caser [Tang and Wang, 2018].
  - NextItNet [Yuan et al., 2019].
  - SRGNN [Wu et al., 2019].
  - SASRec [Kang and McAuley, 2018].

Baselines (2/2)

- Three SHOCCF algorithms:
  - RLBL [Liu et al., 2017] is composed of an RNN module and a log-bilinear (LBL) module, where the former and the latter are used to model the long-term interests and the short-term interests, respectively.
  - RIB [Zhou et al., 2018b] concatenates the item embedding with the behavior embedding, and generates the sequential feature through a GRU layer and an attention layer.
  - BINN [Li et al., 2018] captures the representation of the current consumption motivations by a contextual LSTM (CLSTM), and learns a user's historical stable preferences with a bidirectional CLSTM (Bi-CLSTM).

Parameter Settings (1/2)

- For all the datasets, we set the sequence length L to 50 [Kang and McAuley, 2018].
- For the common parameters of all the methods, we set the embedding size d to 64 [Yuan et al., 2019], the batch size to 128, and the learning rate to 0.001 [Kang and McAuley, 2018].
- The dropout rate for ML1M is 0.2, and 0.5 for the other three datasets [Kang and McAuley, 2018].
- For Caser, the number of vertical filters and horizontal filters are set to 4 and 16, respectively, and the height of the horizontal filters is chosen in the range of $\{2, 3, 4\}$. Besides, the hyperparameters on the regularization are chosen from $\{0.0001, 0.001, 0.01, 0.1\}$ [Tang and Wang, 2018].

Parameter Settings (2/2)

- For NextItNet, the dilation factors are $\{1, 2, 1, 2, 1, 2\}$ according to the code provided by the authors [Yuan et al., 2019].
- For SRGNN, the depth of GNN is set to 1.
- For SASRec, the number of heads in self-attention is set to 1, and the number of blocks is set to 2.
- For the RNN-based methods GRU4Rec, RLBL, RIB and BINN, we set the size of the recurrent unit to 64.
- In the training phase, we use Adam as the optimizer to train all the models, and stop training when the performance on the validation data does not increase for 20 consecutive iterations [Kang and McAuley, 2018].

Main Results (1/3)

Table: Recommendation performance of five SOCCF methods (i.e., GRU4Rec, Caser, NextItNet, SRGNN and SASRec), three SHOCCF methods (i.e., RLBL, RIB and BINN), as well as our BAR with SASRec as the backbone model, on four datasets of ML1M, Rec15, UB and Tmall. The best results are marked in bold, and the second best results are underlined.

Method	ML1M		Rec15		UB		Tmall	
	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
GRU4Rec	0.1249	0.0600	0.3588	0.1921	0.0571	0.0320	0.0744	0.0431
Caser	0.1215	0.0608	0.3985	0.2042	0.0562	0.0289	0.0390	0.0207
NextItNet	0.1236	0.0608	0.3414	0.1816	0.0442	0.0239	0.0718	0.0432
SRGNN	0.1267	0.0614	0.4211	<u>0.2291</u>	0.0575	0.0307	0.0707	0.0419
SASRec	0.1350	0.0651	0.3615	0.1889	<u>0.0744</u>	<u>0.0412</u>	<u>0.0862</u>	<u>0.0521</u>
RLBL	<u>0.1474</u>	<u>0.0729</u>	0.3899	0.2073	0.0462	0.0252	0.0678	0.0405
RIB	0.1302	0.0646	0.3668	0.1921	0.0660	0.0370	0.0776	0.0454
BINN	0.1330	0.0656	<u>0.4333</u>	**0.2299**	0.0646	0.0359	0.0823	0.0487
BAR	**0.1520**	**0.0754**	**0.4358**	0.2176	**0.0760**	**0.0420**	**0.0899**	**0.0538**

Main Results (2/3)

We can have the following observations:

- On average of the performance on the four datasets, our BAR improves 10.43% in terms of HR@10 compared with the most competitive baseline BINN, which clearly demonstrates the superiority of our proposed behavior-aware recommendation framework.
- On the one hand, it shows that our BAR can fully utilize the behavior information without changing the internal implementation of the representation module while serving as a general framework.

Main Results (3/3)

- SASRec performs the best on three of the four datasets, i.e., ML1M, UB and Tmall, which indicates that the self-attention module in SASRec has an advantage when dealing with the heterogeneous behaviors in SHOCCF.
- RIB and BINN defeat four SOCCF methods (i.e., GRU4Rec, Caser, NextItNet and SRGNN) on ML1M, UB and Tmall, which shows the complementarity of the two different types of behaviors in SHOCCF, and jointly modeling them is beneficial.

Ablation Study (1/2)

Table: Ablation studies of our BAR with five different backbone models (i.e., GRU4Rec, Caser, NextItNet, SRGNN and SASRec) on four datasets of ML1M, Rec15, UB and Tmall.

Model	ML1M		Rec15		UB		Tmall	
	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
GRU4Rec	0.1249	0.0600	0.3588	0.1921	0.0571	0.0320	0.0744	0.0431
GRU4Rec+task	0.1391	0.0674	0.3706	0.1982	0.0578	0.0314	0.0784	0.0459
GRU4Rec+att	0.1346	0.0672	0.3930	**0.1999**	0.0564	0.0313	0.0800	**0.0500**
GRU4Rec+BAR	**0.1492**	**0.0752**	**0.3981**	0.1992	**0.0584**	**0.0322**	**0.0807**	0.0497
Caser	0.1215	0.0608	0.3985	0.2042	0.0562	0.0289	0.0390	0.0207
Caser+task	0.1322	0.0651	0.4088	0.2076	0.0582	0.0306	0.0461	0.0245
Caser+att	0.1369	0.0681	0.4253	0.2109	0.0593	0.0322	0.0664	0.0378
Caser+BAR	**0.1495**	**0.0738**	**0.4446**	**0.2230**	**0.0607**	**0.0331**	**0.0682**	**0.0391**
NextItNet	0.1236	0.0608	0.3414	0.1816	0.0442	0.0239	0.0718	0.0432
NextItNet+task	0.1332	0.0659	0.3443	0.1830	0.0434	0.0241	0.0728	0.0437
NextItNet+att	0.1353	0.0650	**0.3648**	**0.1893**	**0.0506**	**0.0273**	0.0739	0.0454
NextItNet+BAR	**0.1396**	**0.0693**	0.3620	0.1877	0.0482	0.0265	**0.0761**	**0.0465**
SRGNN	0.1267	0.0614	0.4211	0.2291	0.0575	0.0307	0.0707	0.0419
SRGNN+task	**0.1373**	**0.0673**	0.4298	0.2368	0.0592	0.0318	0.0765	0.0453
SRGNN+att	0.1251	0.0604	**0.4343**	**0.2423**	0.0540	0.0292	0.0772	**0.0467**
SRGNN+BAR	0.1260	0.0616	0.4279	0.2356	**0.0602**	**0.0340**	**0.0787**	0.0458
SASRec	0.1350	0.0651	0.3615	0.1889	0.0744	0.0412	0.0862	0.0521
SASRec+task	0.1446	0.0716	0.3698	0.1912	0.0738	0.0411	0.0890	0.0521
SASRec+att	0.1421	0.0709	0.4336	**0.2186**	0.0739	0.0411	0.0880	0.0523
SASRec+BAR	**0.1520**	**0.0754**	**0.4358**	0.2176	**0.0760**	**0.0420**	**0.0899**	**0.0538**

Ablation Study (2/2)

We have the following observations:

- **"baseline+task".** Each backbone model knows nothing about the behavior information of the next item, while our BAR informs its real next behavior (i.e., the task-specific layer) during the training phase, which helps the model learn how to distinguish different behaviors.
- **"baseline+att".** Our BAR further learns different weights on the historically interacted items according to the behaviors of the items at different positions (i.e., the behavior attention layer), so that the method can better model the heterogeneous behaviors of users towards the items.
- **"baselin+BAR".** On average of the performance on the four datasets, our BAR improves GRU4Rec, Caser, NextItNet, SRGNN and SASRec by 10.27%, 29.40%, 8.46%, 4.22% and 9.92% in terms of HR@10, respectively, which again shows the advantage of our BAR.

Exploration Experiments (1/3)

- We divide the test data into two separate sets in order to gain some deep understanding, i.e., an unexamined test set and an examined test set, according to the fact whether the items in the test data have been examined by the corresponding user during the training phase.

Exploration Experiments (2/3)

Table: Recommendation performance of GRU4Rec, Caser, NextItNet, SRGNN, SASRec and their improved versions on the examined test set and the unexamined test set of UB and Tmall.

Model	UB (examined, 21.1%)		UB (unexamined, 78.9%)		Tmall (examined, 12.96%)		Tmall (unexamined, 87.04%)	
	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
GRU4Rec	0.2068	**0.1188**	0.0171	0.0088	0.3637	0.2280	0.0313	0.0155
GRU4Rec+task	0.2056	0.1155	**0.0182**	0.0090	0.3758	0.2362	**0.0341**	**0.0175**
GRU4Rec+att	0.2011	0.1147	0.0177	0.0089	0.3969	**0.2741**	0.0328	0.0166
GRU4Rec+BAR	**0.2095**	0.1185	0.0180	**0.0092**	**0.4045**	0.2705	0.0324	0.0169
Caser	0.2218	0.1187	0.0120	0.0048	0.1561	0.0878	0.0216	0.0107
Caser+task	0.2236	0.1214	0.0139	0.0063	0.1839	0.1048	0.0256	0.0126
Caser+att	0.2261	0.1270	0.0147	0.0069	**0.3265**	**0.2023**	0.0276	0.0133
Caser+BAR	**0.2270**	**0.1284**	**0.0163**	**0.0076**	0.3202	0.1989	**0.0307**	**0.0154**
NextItNet	0.1550	0.0849	0.0146	0.0076	0.3453	0.2269	0.0311	0.0159
NextItNet+task	0.1516	0.0853	0.0145	0.0077	0.3350	0.2234	**0.0337**	**0.0170**
NextItNet+att	**0.1788**	**0.0991**	**0.0163**	0.0081	**0.3664**	0.2454	0.0304	0.0156
NextItNet+BAR	0.1675	0.0927	**0.0163**	**0.0088**	**0.3664**	**0.2481**	0.0328	0.0165
SRGNN	0.2100	0.1157	0.0168	0.0080	0.3430	0.2237	0.0302	0.0148
SRGNN+task	**0.2240**	0.1218	0.0151	0.0078	0.3700	0.2371	0.0328	0.0167
SRGNN+att	0.2025	0.1108	0.0143	0.0073	**0.4027**	**0.2696**	0.0287	0.0136
SRGNN+BAR	0.2224	**0.1273**	**0.0168**	**0.0091**	0.3744	0.2403	**0.0346**	**0.0169**
SASRec	0.2861	0.1640	0.0177	0.0083	0.4462	0.2952	0.0326	0.0159
SASRec+task	0.2733	0.1565	**0.0204**	**0.0102**	0.4413	0.2828	**0.0366**	**0.0178**
SASRec+att	0.2886	0.1642	0.0165	0.0082	0.4448	0.2900	0.0349	0.0169
SASRec+BAR	**0.2979**	**0.1701**	0.0166	0.0078	**0.4520**	**0.2974**	0.0360	0.0175

Exploration Experiments (3/3)

We have the following observations:

- The task-specific layer focuses on improving the performance of the unexamined test set, while the behavior attention layer is more suitable to promote the performance on the examined test set.
- With the help of the behavior attention layer and the task-specific layer, our BAR can improve the performance of both parts of the data, so that our baseline+BAR can outperform baseline+task and baseline+att, as well as the baseline, in most cases.

Fusion of Behavior Information (1/3)

- We compare baseline+BAR with baseline+plus, which introduces the behavior information by adding the behavior embedding to the item embedding directly, i.e.,
  $h_{t-1} = RM(V_{i_u^{t-L}} + B_{b_u^{t-L}}, \ldots, V_{i_u^{t-1}} + B_{b_u^{t-1}})$.

Fusion of Behavior Information (2/3)

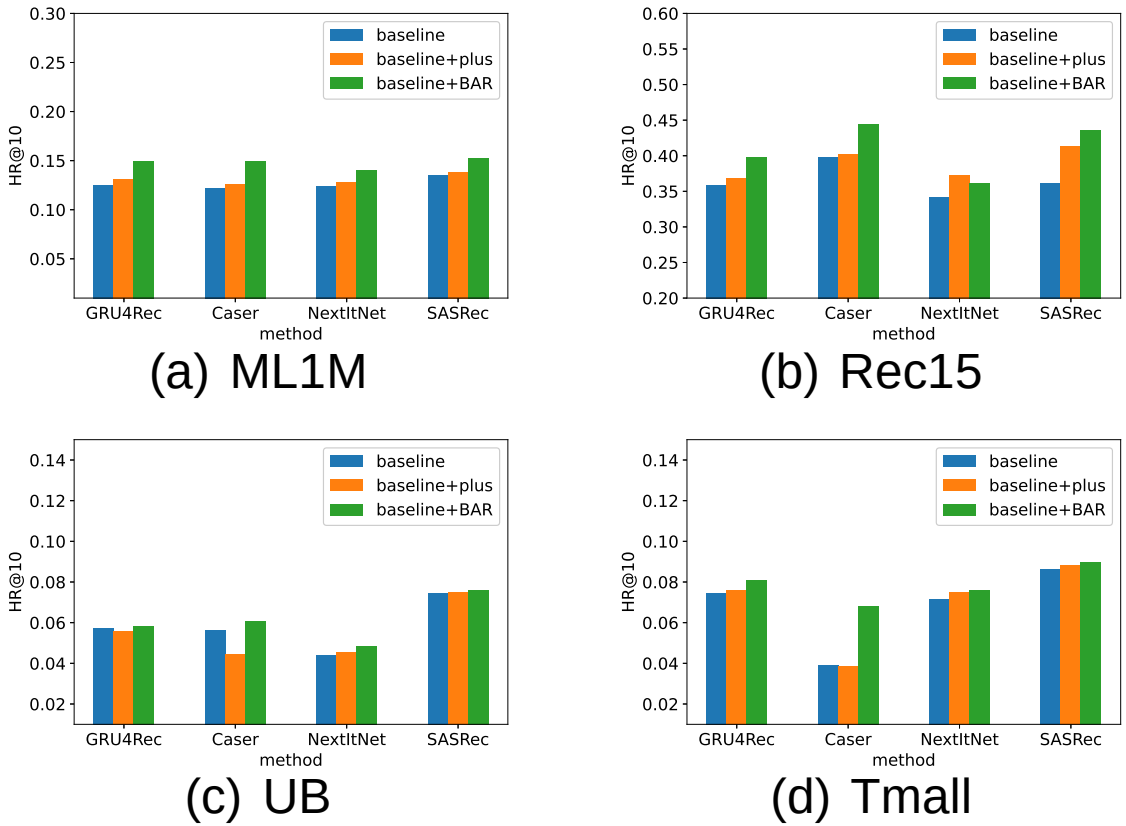


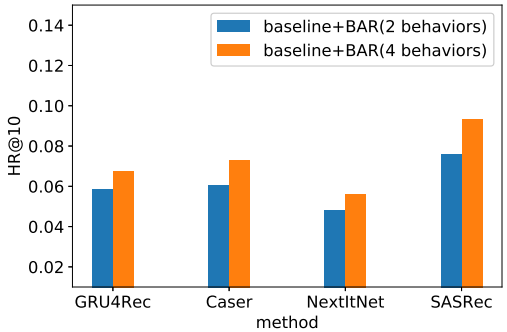
Figure: Recommendation performance of each baseline and the corresponding baseline+plus and baseline+BAR on ML1M, Rec15, UB and Tmall.

Fusion of Behavior Information (3/3)

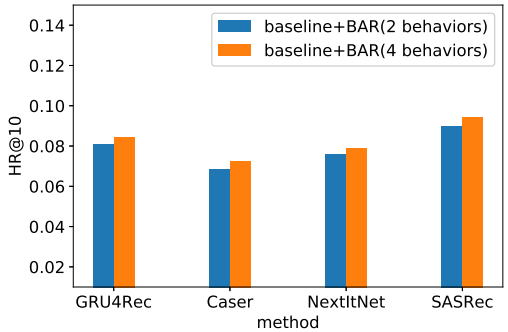
We have the following observations:

- baseline+plus improves the performance of the baseline model in most cases, showing that the introduction of the behavior information can help the baseline method to better model users' interests.
- The performance of Caser+plus is reduced by 19.06% compared with that of Caser on UB, indicating that fusing the behavior information via addition is a more radical way, which may interfere with the training of the corresponding baseline.
- Our BAR has a stable improvement over all the methods and the greatest improvement over the corresponding baseline in most cases. This shows that the way our BAR adopting an indirect method to introduce the behavioral information is a more reasonable solution. It avoids direct interference to the space of the input vector by increasing or reducing the weight of the items in the input sequence.

Extension with More Behavior Types (1/2)



(a) UB



(b) Tmall

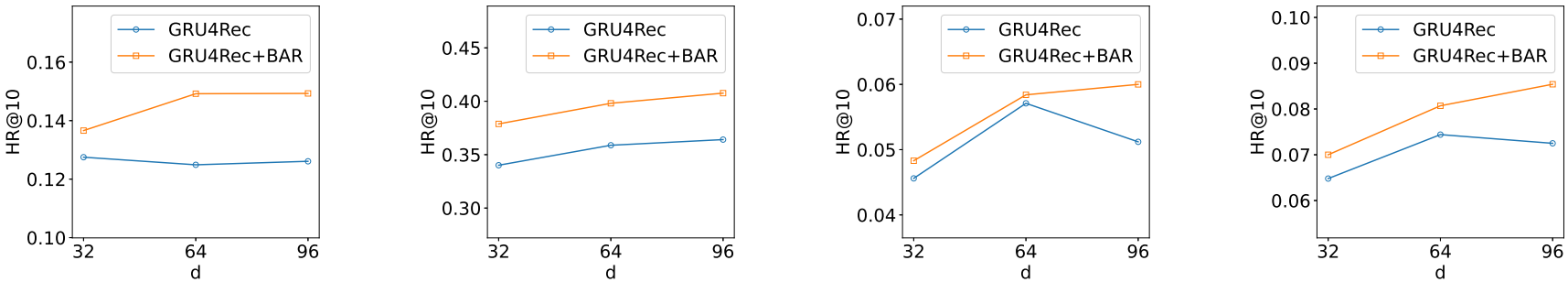
Figure: Recommendation performance of each baseline+BAR with two types of behaviors and four types of behaviors, denoted as 'baseline+BAR(2 behaviors)' and 'baseline+BAR(4 behaviors)', respectively, on UB and Tmall.

Extension with More Behavior Types (2/2)

We have the following observations:

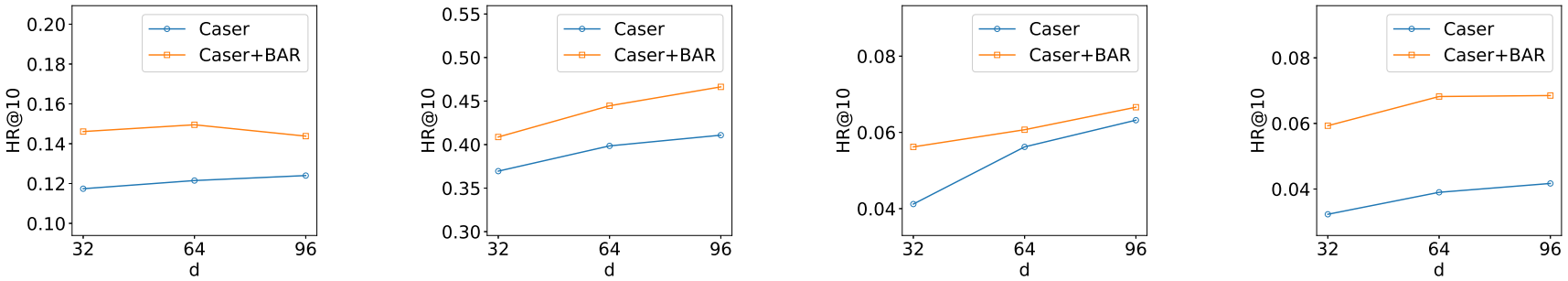
- each baseline+BAR (4 behaviors) achieves better performance than the corresponding baseline+BAR (2 behaviors) in all cases, which shows that more types of behaviors can help the task-specific layer and the behavior attention layer to better capture users' preferences.
- Our BAR can easily be extended to scenarios with more than two types of behaviors without changing the model structure, which shows its usefulness and flexibility.

Sensitivity of Embedding Dimensionality (1/3)



(a) ML1M (b) Rec15 (c) UB (d) Tmall

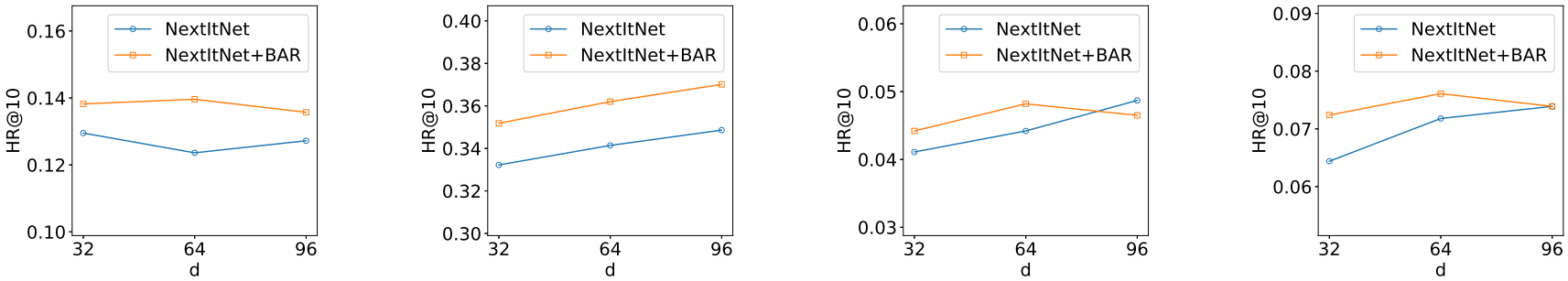
Figure: Recommendation performance of GRU and GRU+BAR with different values of dimensionality d on four datasets.



(a) ML1M (b) Rec15 (c) UB (d) Tmall

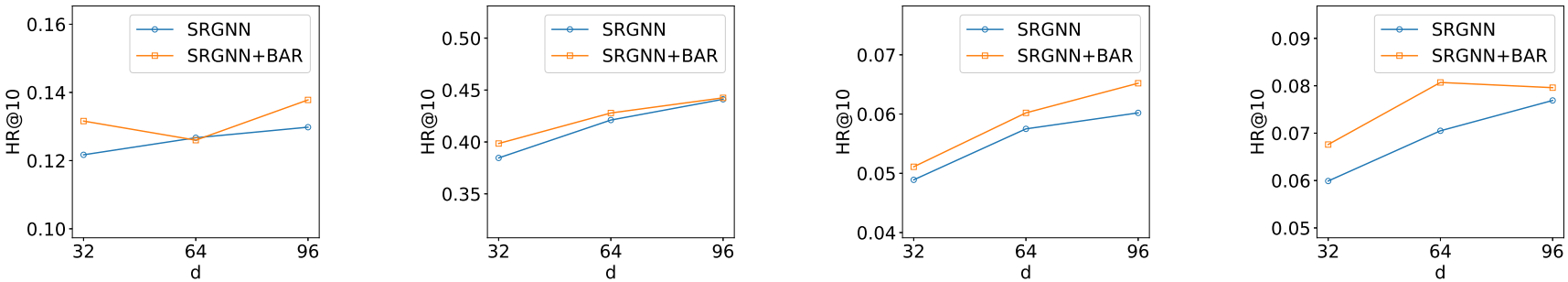
Figure: Recommendation performance of Caser and Caser+BAR with different values of dimensionality d on four datasets.

Sensitivity of Embedding Dimensionality (2/3)



(a) ML1M (b) Rec15 (c) UB (d) Tmall

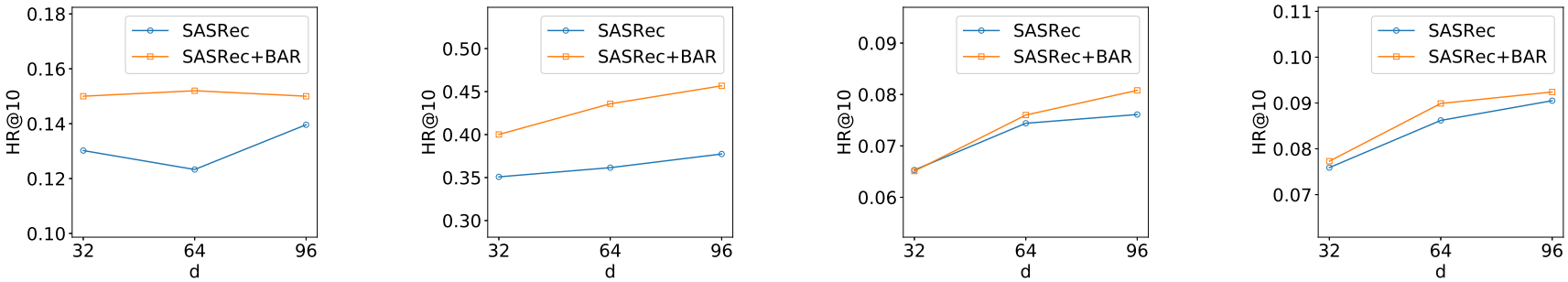
Figure: Recommendation performance of NextItNet and NextItNet+BAR with different values of dimensionality *d* on four datasets.



(a) ML1M (b) Rec15 (c) UB (d) Tmall

Figure: Recommendation performance of SRGNN and SRGNN+BAR with different values of dimensionality *d* on four datasets.

Sensitivity of Embedding Dimensionality (3/3)



(a) ML1M (b) Rec15 (c) UB (d) Tmall

Figure: Recommendation performance of SASRec and SASRec+BAR with different values of dimensionality *d* on four datasets.

We have the following observations:

- Our BAR achieves better performance with different values of the dimensionality *d* for all the baseline methods, which clearly shows that our BAR has a strong adaptability for an SOCCF method to be applied to the SHOCCF problem.

Conclusions

- We propose a generic framework BAR to adapt a typical sequential recommendation method from SOCCF to SHOCCF.
- We design a behavior attention layer to model the relationship between the target behavior and the current behaviors, and then add the attention weight as a correction factor to the historical item sequence, so that the model can digest different behaviors in the sequence better.
- We design a task-specific layer, which combines the output feature with the real behavior at the next timestamp as prior knowledge so that the model can distinguish different tasks.
- Extensive empirical studies on four public datasets show that our BAR can significantly improve the performance of a typical sequential recommendation method for the studied SHOCCF problem.

Future Work

In the future, we are interested in further studying and improving our BAR in scenarios with richer information, including (i) knowledge graph [Chen et al., 2021] of the items and their attributes and (ii) social connections among the users [Zhao et al., 2020, Liao et al., 2022].

Thank you!

- We thank the Editor in Chief, the handling associate editor and reviewers for their efforts and constructive expert comments, Ms. Wanqi Ma for her help on linguistic quality improvement, and the support of National Natural Science Foundation of China Nos. 62172283 and 61836005.

- If you have any questions, please feel free to contact us.

Chen, J., Yu, J., Lu, W., Qian, Y., and Li, P. (2021).

IR-Rec: An interpretive rules-guided recommendation over knowledge graph.
Information Sciences, 563:326–341.

Hidasi, B. and Karatzoglou, A. (2018).

Recurrent neural networks with top-k gains for session-based recommendations.
In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 843–852.

Kabbur, S., Ning, X., and Karypis, G. (2013).

FISM: Factored item similarity models for top-N recommender systems.
In *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 659–667.

Kang, W. and McAuley, J. J. (2018).

Self-attentive sequential recommendation.
In *Proceedings of the 18th IEEE International Conference on Data Mining*, pages 197–206.

Li, Z., Zhao, H., Liu, Q., Huang, Z., Mei, T., and Chen, E. (2018).

Learning from history and present: Next-item recommendation via discriminatively exploiting user behaviors.
In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1734–1743.

Liang, D., Krishnan, R. G., Hoffman, M. D., and Jebara, T. (2018).

Variational autoencoders for collaborative filtering.
In *Proceedings of the 27th International Conference on World Wide Web*, WWW'18, pages 689–698.

Liao, J., Zhou, W., Luo, F., Wen, J., Gao, M., Li, X., and Zeng, J. (2022).

SocialLGN: Light graph convolution network for social recommendation.
Information Sciences, 589:595–607.

Liu, Q., Wu, S., and Wang, L. (2017).

Multi-behavioral sequential prediction with recurrent log-bilinear model.
IEEE Transactions on Knowledge and Data Engineering, 29(6):1254–1267.

Pan, W., Yang, Q., Cai, W., Chen, Y., Zhang, Q., Peng, X., and Ming, Z. (2019).

Transfer to rank for heterogeneous one-class collaborative filtering.
ACM Transactions on Information Systems, 37(1):10:1–10:20.

Tang, J. and Wang, K. (2018).

Personalized top-N sequential recommendation via convolutional sequence embedding.
In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, pages 565–573.

Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., and Tan, T. (2019).

Session-based recommendation with graph neural networks.
In *Proceedings of the 33th AAAI Conference on Artificial Intelligence*, pages 346–353.

Yuan, F., Karatzoglou, A., Arapakis, I., Jose, J. M., and He, X. (2019).

A simple convolutional generative network for next item recommendation.
In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining*, pages 582–590.

Zhao, W., Ma, H., Li, Z., Ao, X., and Li, N. (2020).

Improving social and behavior recommendations via network embedding.
Information Sciences, 516:125–141.

Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H., and Gai, K. (2018a).

Deep interest network for click-through rate prediction.
In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1059–1068.

Zhou, M., Ding, Z., Tang, J., and Yin, D. (2018b).

Micro behaviors: A new perspective in e-commerce recommender systems.
In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, pages 727–735.