

Session-based Recommendation with Hypergraph Attention Networks

Jianling Wang*

Kaize Ding†

Ziwei Zhu*

James Caverlee*

Abstract

Session-based recommender systems aim to improve recommendations in short-term sessions that can be found across many platforms. A critical challenge is to accurately model user intent with only limited evidence in these short sessions. For example, is a flower bouquet being viewed meant as part of a wedding purchase or for home decoration? Such different perspectives greatly impact what should be recommended next. Hence, this paper proposes a novel session-based recommendation system empowered by hypergraph attention networks. Three unique properties of the proposed approach are: (i) it constructs a hypergraph for each session to model the item correlations defined by various contextual windows in the session simultaneously, to uncover item meanings; (ii) it is equipped with hypergraph attention layers to generate item embeddings by flexibly aggregating the contextual information from correlated items in the session; and (iii) it aggregates the dynamic item representations for each session to infer the general purpose and current need, which is decoded to infer the next interesting item in the session. Through experiments on three benchmark datasets, we find the proposed model is effective in generating informative dynamic item embeddings and providing more accurate recommendations compared to the state-of-the-art.

1 Introduction

Recommendation systems are ubiquitous, acting as an essential component in online platforms to help users discover items of interest. In many practical scenarios, a recommendation system needs to infer the next interaction for a user based only on the short-term prior interactions within a particular session. Previous approaches that learn a static model of users [10, 11, 22, 31, 34] or rely on long-term user behaviors [9, 17, 28, 30] are not well-suited for such scenarios and could lead to poor predictive power. In contrast, *session-based recommendation* has attracted increasing attention, with recent approaches showing promising performance in inferring user interests in these (often anonymous) short-term sessions [16, 19, 21, 23, 32].

A critical issue is how items are treated in such session-based recommendation approaches. The *individual* items can reveal user intent, but they only provide limited evidence. To illustrate, consider the example in Figure 1. In different sessions, the same flower bouquet can be viewed differently, i.e., as part of a wedding party purchase, an option for home decoration, or

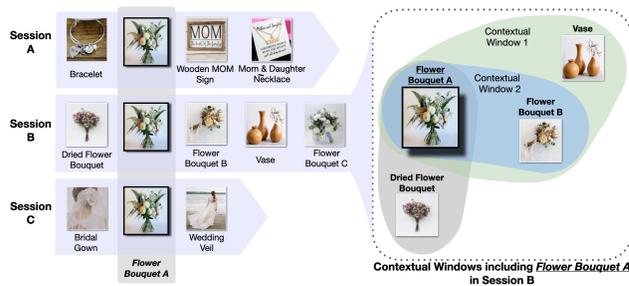


Figure 1: An example with three sessions. The same flower bouquet appears in each session, but with a different purpose in each. The righthand side shows contextual windows of size 2 and 3 including this flower bouquet. Items in the same contextual window are correlated; and different contextual windows may have different levels of importance for characterizing an item.

a gift for Mother’s Day. However, if we independently consider Flower Bouquet A, it might be viewed as exactly the same item across sessions. In a sense, the meaning of an item (and what it reveals about user intent) could be inferred from *contextual windows*, each of which contains a set of consecutive items showing up together within a session. In Session C, bouquet A is likely for a wedding party since the user clicks it right between other wedding-related items, while the bouquet in Session A may be a Mother’s Day gift since it is clicked along with items specified for “Mom.”

Hence, we propose to exploit these contextual windows to model *session-wise item representations* that can robustly capture user intent with only limited evidence available in short sessions. However, there are several key challenges in eliciting the user intent signal among items from the contextual windows in each session: (1) conventional graph structures and graph neural networks [2, 3, 7, 8, 15] are designed to model the pairwise connections between items, which are not always sufficient since we need to consider contextual windows connecting various numbers of items ranging from two to many. For example, for Session B in Figure 1, the pairwise linkage between bouquet A and B is not enough to reveal that bouquet A is meant for home decoration. But such evidence could be inferred by analyzing the triadic relations among both the bouquets

*Texas A&M University, {jllwang, zhuziwei, caverlee}@tamu.edu

†Arizona State University, kaize.ding@asu.edu

and a vase as defined by Contextual Window 1. To this end, we adopt the hypergraph [1, 4, 6, 29] structure to model the correlations amongst items in different contextual windows within each session. In a session hypergraph, each node denotes an item, and a hyperedge connects the collection of items that show up together within a contextual window. This hypergraph structure supports capturing correlations among items defined by contextual windows, which could be arbitrary-order depending on the usage scenario; **(2)** while propagating and aggregating the user intent evidence within a session hypergraph, some items are informative, but others may not be. Moreover, different contextual windows may bring in different levels of evidence for how items are represented. Thus, a key challenge is how to carefully highlight the informative items on each hyperedge and also emphasize the evidence from hyperedges with larger impacts.

To tackle the aforementioned challenges, we propose SHARE: a **S**ession-based **H**ypergraph **A**ttention Network for **R**ecommendation. Specifically, the proposed SHARE has three unique characteristics:

- First, it captures contextual information with sliding windows, whereby items appearing in the same window are connected with a hyperedge. By applying multiple sliding windows on the session sequence, it is able to model a session considering contextual windows of varying sizes simultaneously in the hypergraph structure.
- Second, it incorporates a carefully-designed hypergraph attention network to extract the user intent evidence from contextual windows, which is able to pay more attention on the informative items (nodes) and also emphasize the evidence from contextual windows (hyperedges) with larger impacts.
- Third, the session-wise item embeddings resulting from a stack of hypergraph attention layers can then be fed into a self-attention layer to infer both the general intent and current interests in the session, which are decoded jointly to generate the next-item recommendation for this session.

With experiments on three benchmark datasets for session-based recommendation, we show that such a hypergraph-based approach is better suited than conventional graphs for modeling item correlations within sessions. The proposed SHARE is effective in predicting the next interesting items, and significantly outperforms the state-of-the-art in session-based recommendation.

2 Related Work

Session-based Recommendation. Given sequential user behaviors in a session, session-based recommen-

dation systems aim to infer the subsequent behavior. Since these sessions usually exclude login information and take place in the short-term (e.g., less than a half hour) where user intent is local to the session, previous works that elicit user preferences from long-term historic behaviors or generate user embeddings based on their identification [11, 28, 30] are not well-suited to providing recommendations for these anonymous short-term sessions. Though matrix factorization-based and neighbor-based methods can be applied for session-based recommendation [18, 24], they ignore the transitional patterns between items. Thus, FPMC [23] has been proposed to extend matrix factorization with Markov Chains, which is able to predict the next item considering the transition from the last item. With the advancement of neural networks, different neural mechanisms have been applied to handle sequential session data. In [12, 13], the authors adopt recurrent neural networks to learn sequential patterns from all sessions and infer the next items with the output of the last layer. NARM [16] utilizes an attention layer to aggregate items in the session and capture the main purpose of each session. Furthermore, STAMP [19] designs another attention component to emphasize the short-term interest in the session. To capture the local dependencies between items, SR-GNN [32] applies the conventional graph neural network to model the transitions between items in a session graph. In contrast, we propose to utilize hypergraphs and an attention mechanism to model item correlations and user intent with the contextual windows in session-based recommendation.

Graph-based Recommendation. Since graphs can be a good fit for modeling the interactions between item-item, user-user, or user-item, there are works in developing recommendation systems centering around different graph structures. Translation-based recommendation systems [9, 20] treat all the items as nodes and users as the connections between items consumed subsequently. These models embed users and items into a similar space by minimizing the translation loss, which requires clear user identification (which is often not available in session-based scenarios). Recently, several efforts apply newly introduced Graph Neural Networks (GNNs) for recommendation [2, 31, 33], in which different GNN models have been designed for representation learning in graph structured data. GCMC [2] consists of neural graph autoencoders to reconstruct the user-item rating graph. And in NGCF [31], the authors propose to construct a user-item bipartite graph and utilize multiple graph neural layers to capture the multi-hop collaborative signals between users and items. In social recommendation, the relations between users can be exploited

with the social graph [5, 32]. For example, DiffNet [5] adopts a Graph Convolutional Network to model the diffusion of user embeddings among their social connections. Unlike SHARE, these methods are built on conventional graph structures (not hypergraphs). And they are not designed for the special characteristics of session-based recommendation, like sequential user actions in short-lived sessions without user identification.

3 The Proposed Model - SHARE

In this section, we propose a novel session-based recommendation model with Hypergraph Attention Networks to exploit the contextual windows within each individual session. In the following, we begin with the problem setup and then structure the design of SHARE around three research questions: (i) Given that items falling into the same contextual window are correlated, how do we construct a hypergraph for each session to model the correlations among items from a variety of contextual windows simultaneously? (ii) Considering these contextual windows, how do we update an item embedding with the user intent evidence that propagates in and across the contextual windows? (iii) With these session-wise item embeddings, how can we infer the next interesting item by extracting both the general interest and current need in the session?

3.1 Problem Setup. In session-based recommendation [16, 21, 32], given the sequence of items which have been interacted within a session, the goal is to predict the next item. Let $\mathbf{I} = \{i_1, i_2, \dots, i_N\}$ denote the set of N unique items in the system. These items start with a set of embeddings $\{\mathbf{i}_1, \mathbf{i}_2, \dots, \mathbf{i}_N\}$, each of which is a trainable embedding associated with the item ID. An anonymous session s consisting of a sequence of t actions (i.e., the items interacted within the session) can be denoted by $\mathbf{s} = [i_{s,1}, i_{s,2}, \dots, i_{s,t}]$, in which the items are sorted in chronological order and $i_{s,p}$ represents the p th item interacted within the session. A session-based recommendation system should predict the next possible action (i.e., $i_{s,t+1}$) based on the previous t actions. That is, we want to generate the preference scores for all the items in \mathbf{I} based on the sequence of actions in session s . Then the top-K preferred items can be treated as candidates for recommendation.

3.2 Session Hypergraph Construction. With a sequence of interacted items in session s , we propose to model the session as a hypergraph before exploiting the item correlations in different (possibly, overlapping) contextual windows. Let $\mathcal{G}_s = \{\mathcal{V}_s, \mathcal{E}_s\}$ denote the hypergraph constructed from session s , in which the node set \mathcal{V}_s consists of all the unique items appearing in

this session. Each hyperedge $e \in \mathcal{E}_s$ will connect all the items falling into the specific contextual window. As in Figure 2, we can apply a sliding window with size w on the item sequence for the session to identify all of the contextual windows of size w in this session, with which the items appearing in the same window can be treated as items falling into the same contextual window and thus will be connected with a hyperedge. Hence, to exploit contextual windows of varying sizes in the session, we can apply sliding windows of varying sizes on the item sequence. Let \mathcal{E}_s^w represent the collection of all the hyperedges constructed with such a sliding window of size w on session s . Then we gather hyperedges based on different sliding windows together to be the set of hyperedges \mathcal{E}_s for session s with $\mathcal{E}_s = \mathcal{E}_s^2 \cup \mathcal{E}_s^3 \cup \dots \mathcal{E}_s^W$, in which W is the maximum size of contextual windows that we consider in the model. We repeat this process to construct the unique hypergraph for each of the sessions.

3.3 Hypergraph Attention Networks. With the hypergraph structure, we want to learn the representations for nodes (items) considering the correlations among nodes (items) defined by different hyperedges. In the following, we introduce a novel hypergraph attention network (HGAT) which is able to aggregate the user intent evidence that propagates in and across contextual windows while updating the node representations in a hypergraph. There are recent works extending neural networks from a conventional graph to a hypergraph by generalizing the convolution operation [1, 6, 29], with which the neighboring node features will be firstly aggregated to the common hyperedges and then propagated to the node. They usually treat nodes *equally* while characterizing the hyperedge or use a pre-defined weight matrix to model the importance of information propagated via different hyperedges. However, in session hypergraphs, for nodes belonging to the same contextual window, some of them may be informative in revealing user intent, but others may not be. And the evidence propagating via different hyperedges (contextual windows) may bring in different levels of impacts to an item. Thus, we require a new representation learning process for hypergraphs to capture these special needs.

To tackle this problem, we propose to generalize the attention mechanism for hypergraphs so that we can *highlight the informative items on each hyperedge and also emphasize the evidence from hyperedges with larger impacts*. In the following, we will explain the node representation learning process with the proposed hypergraph attention layer (see Figure 2) in two steps:

Node to Hyperedge. Since information can propagate among neighboring nodes via hyperedges, they are the key factor for node representation learning in hyper-

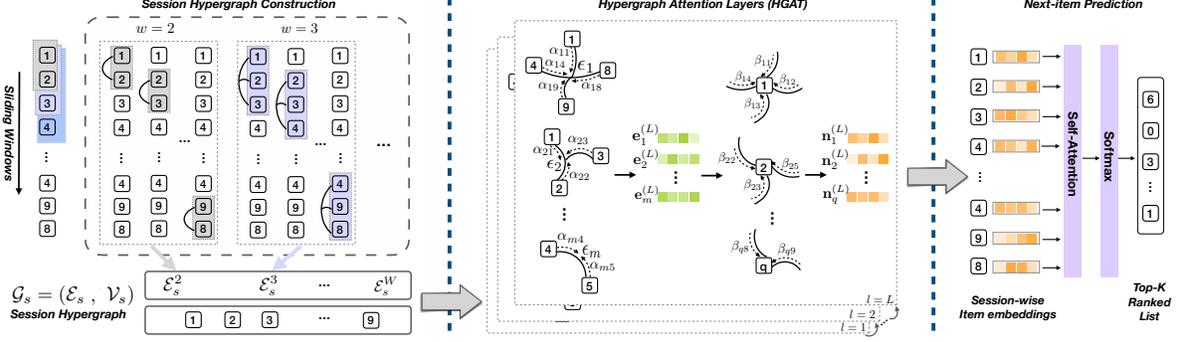


Figure 2: The structure of SHARE: It applies multiple sliding windows to capture the contextual information to construct a hypergraph. With a well-designed HGAT network, it is able to generate the item embeddings revealing their meanings in the specific session. The sequence of session-wise item embeddings is fed into the self-attention layer to generate an embedding for the session, which is decoded for the preference scores on items.

graph. With the special hypergraph structure, instead of directly updating each node with the the neighboring node information, we firstly treat each hyperedge as an interlayer between nodes and aggregate all the information propagating via the hyperedge.

We take the graph \mathcal{G}_s constructed by session s as an example. The operation can be applied on hypergraphs for other sessions. Let $\{\mathbf{n}_1^{(0)}, \mathbf{n}_2^{(0)}, \dots, \mathbf{n}_p^{(0)}\} = \{\mathbf{i}_1, \mathbf{i}_2, \dots, \mathbf{i}_p\}$ denote the node input for the first HGAT layer, which is the initial item embeddings for the set of unique items $\{i_1, i_2, \dots, i_p\}$ appearing in \mathcal{G}_s . Since some nodes on a hyperedge are informative but others may not be, we should pay varying attention on the information from these nodes while aggregating them together. Let $\mathbf{m}_{t \sim j}^{(1)}$ denote the information propagating via hyperedge ϵ_j from node t on the $(1)_{th}$ HGAT layer. We can aggregate the information from each of the nodes connected by ϵ_j with the attention operation to generate the representation $\mathbf{e}_j^{(1)}$ as:

$$(3.1) \quad \mathbf{e}_j^{(1)} = \sum_{t \in \mathcal{N}_j} \mathbf{m}_{t \sim j}^{(1)} \quad \mathbf{m}_{t \sim j}^{(1)} = \alpha_{jt} \mathbf{W}_1^{(1)} \mathbf{n}_t^{(0)}$$

$$\alpha_{jt} = \frac{S(\hat{\mathbf{W}}_1^{(1)} \mathbf{n}_t^{(0)}, \mathbf{u}^{(1)})}{\sum_{f \in \mathcal{N}_j} S(\hat{\mathbf{W}}_1^{(1)} \mathbf{n}_f^{(0)}, \mathbf{u}^{(1)})}$$

in which \mathcal{N}_j denotes all the nodes connected by hyperedge ϵ_j and $\mathbf{u}^{(1)}$ represents a trainable node-level context vector for the $(1)_{th}$ HGAT layer. $\mathbf{W}_1^{(1)}$ and $\hat{\mathbf{W}}_1^{(1)}$ are the transform matrices and α_{jt} denotes the attention score of node t on hyperedge ϵ_j . We use a function $S(\cdot, \cdot)$ to calculate the similarity between the node embedding and context vector. Empirically, we use *Scaled Dot-Product Attention* to calculate the attention scores [14, 26], which is defined as:

$$(3.2) \quad S(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a}^T \mathbf{b}}{\sqrt{D}}$$

where D is the dimension size and can be used for normalization while calculating the similarity scores.

Hyperedge to Node. To update the embedding for a node, we need to aggregate the contents from all its connected hyperedges. Similarly, we utilize the attention mechanism to perform the aggregation in order to model the significance of different hyperedges. Let $\mathbf{m}_{j \rightarrow t}^{(1)}$ denote information (user intent evidence) from hyperedge ϵ_j to node t . Given the set of hyperedges \mathcal{Y}_t which are connected to node t , its update embedding is calculated as:

$$(3.3) \quad \mathbf{n}_t^{(1)} = \sum_{j \in \mathcal{Y}_t} \mathbf{m}_{j \rightarrow t}^{(1)} \quad \mathbf{m}_{j \rightarrow t}^{(1)} = \beta_{tj} \mathbf{W}_2^{(1)} \mathbf{e}_j^{(1)}$$

$$\beta_{tj} = \frac{S(\hat{\mathbf{W}}_2^{(1)} \mathbf{e}_j^{(1)}, \mathbf{W}_3^{(1)} \mathbf{n}_t^{(0)})}{\sum_{f \in \mathcal{Y}_t} S(\hat{\mathbf{W}}_2^{(1)} \mathbf{e}_f^{(1)}, \mathbf{W}_3^{(1)} \mathbf{n}_t^{(0)})}$$

where $\mathbf{W}_2^{(1)}$, $\hat{\mathbf{W}}_2^{(1)}$ and $\mathbf{W}_3^{(1)}$ are the trainable matrices used to transform the vector before calculating the attention scores for the $(1)_{th}$ HGAT layer. β_{tj} indicates the impact of hyperedge ϵ_j on node t . As in the last step, we use the Scaled Dot-Product Attention formula defined by Equation 3.2 to calculate S . The resulting $\mathbf{n}_t^{(1)}$ can be treated as the updated embedding for note t by aggregating information from its neighboring region in the hypergraph.

High-order Propagation. While a single HGAT layer can capture the information from direct neighbors, we construct a Hypergraph Attention Network by stacking multiple HGAT layers to model the multi-hop high-order information propagation in the hypergraph. In this Hypergraph Attention Network, the output of the $(l-1)_{th}$ HGAT layer is the input for the l_{th} layer. Thus the outputs (i.e., node embeddings) from the last layer (L) inherit the contextual information from all the previous layers and can be used to characterize the items

in this session. For each of the session hypergraph, such an Hypergraph Attention Network is able to generate session-wise item embeddings to reflect user intent in the session, by highlighting the informative items on each hyperedge and also emphasizing the evidence from hyperedges with larger impacts.

3.4 Next-Item Prediction. To infer the next interesting item, our goal is to generate an embedding which can encode both the general interest and the current need of the session. We adopt the idea from self-attention [14, 26] to achieve this goal. Since the general interest can be uncovered by the aggregation of all the items and the current need is revealed by the last item, we want to aggregate the items in the session while paying more attention on those items which are highly correlated to the last one. Thus, we can *treat the last item in the session as the query and the sequence of items in the session as both keys and values*, leading to the design of the self-attention layer in Figure 2.

For a session interacted with $[i_{s,1}, i_{s,2}, \dots, i_{s,t}]$ sequentially, we will lookup the corresponding node embeddings from the output of the hypergraph attention networks to get $[\mathbf{n}_{s,1}^{(L)}, \mathbf{n}_{s,2}^{(L)}, \dots, \mathbf{n}_{s,t}^{(L)}]$. We will transform the item embeddings with \mathbf{W}_Q , \mathbf{W}_K and \mathbf{W}_V to generate the query vectors, key vectors, and value vectors correspondingly. Then we can aggregate the sequence of embeddings with:

$$(3.4) \quad \mathbf{h}_s = \sum_{i \leq t} \sigma_{ti} \mathbf{W}_V \mathbf{n}_{s,i}^{(L)}$$

$$\sigma_{ti} = \frac{S(\mathbf{W}_Q \mathbf{n}_{s,t}^{(L)}, \mathbf{W}_K \mathbf{n}_{s,i}^{(L)})}{\sum_{j \leq t} S(\mathbf{W}_Q \mathbf{n}_{s,t}^{(L)}, \mathbf{W}_K \mathbf{n}_{s,j}^{(L)})}$$

in which the attention scores S is defined in Equation 3.2. According to observations in previous research [14, 32], the order of items is less likely to be related to the general interest in short-term sequences and could introduce noise in modeling such short-term sequences. We omit the order information while handling the item sequences in session-based recommendation.

With this carefully-designed self-attention layer, the resulting \mathbf{h}_s encodes both the general interest and the current need for session s . Then we compute the multiplication between \mathbf{h}_s and the latent factor of item v to predict the preference score of session s on v using $p_{sv} = \mathbf{h}_s^T \mathbf{i}_v$. Let $\mathbf{p}_s = [p_{s1}, p_{s2}, \dots, p_{sN}]$ denote the predicted preference scores of session s on all of the N items in the system, which will be processed with a Softmax layer to generate the final scores such that: $\hat{\mathbf{y}}_s = \text{softmax}(\mathbf{p}_s)$. We use a one-hot vector $\mathbf{y}_s = [y_{s1}, y_{s2}, \dots, y_{sN}]$ to denote the ground-truth item of session s . During the training phrase, we calculate

	# Sessions (Training)	# Sessions (Testing)	# Items	Avg. Length of Sessions
<i>YooChoose</i> _{1/64}	369,859	55,898	16,766	6.16
<i>YooChoose</i> _{1/4}	5,917,745	55,898	29,618	5.71
<i>Diginetica</i>	719,470	60,858	43,097	5.12

Table 1: Dataset Statistics.

the cross-entropy loss for all training sessions S_{train} with $\mathcal{L} = -\sum_{s \in S_{train}} \sum_{v=1}^N y_{sv} \log \hat{y}_{sv}$. Then we can train the model using back-propagation. In the testing phase, given an unseen session, we can construct a new hypergraph and feed it into the hypergraph attention network. And then we can calculate its preference scores on all the items and recommend the items ranked among the top.

4 Experiments

In this section, we conduct several experiments to evaluate how the proposed SHARE model performs in session-based recommendation.

4.1 Data. We adopt two public datasets that have been widely used to evaluate session-based recommendation: YooChoose and Diginetica [13, 16, 21, 32]. Yoochoose contains sessions of click events from an online retailer in Europe and was originally published as part of the 2015 RecSys Challenge. Diginetica contains sessions of product transaction data from an online retailer and was released as part of the 2016 CIKM Cup.

We keep sessions with length longer than 1 and items which appear in at least 5 sessions. For Yoochoose, we test on all the sessions happening on the last day in the dataset while training all the sessions before that. Only items appearing in the training set are considered. As for Diginetica, we split the dataset and use the sessions happening in the last 7-days for testing. Furthermore, we adopt a standard sequence preprocessing method used in previous work [19, 25, 32] to generate the session sequences and labels. Then we get the final Diginetica dataset as described in Table 1. However, it is not necessary to train on the entire set of training sequences from YooChoose since it is extremely large and training only on a fraction of recent sessions can lead to better prediction performance based on the experimental results in [25]. As in [16, 19, 32], we sort all of the training sequences generated from YooChoose, and retrieve the most recent 1/64 and 1/4 to be the training samples in *YooChoose*_{1/64} and *YooChoose*_{1/4} (listed in Table 1). Note that *YooChoose*_{1/64} and *YooChoose*_{1/4} share the same set of testing samples. In addition, for fair comparison, the training samples and testing samples in all of the three datasets are exactly the same as in [16, 19, 21, 32].

Datasets		S-Pop	MF	FPMC	KNN	GRU4Rec+	NARM	STAMP	RPNet	SR-GNN	SHARE
<i>YooChoose</i> _{1/64}	Hit@20	30.44	31.31	45.62	51.60	67.84	68.32	68.74	69.13	70.57†	71.51*
	MRR@20	18.35	12.08	15.01	21.81	29.00	28.63	29.67	30.24	30.94†	31.45*
<i>YooChoose</i> _{1/4}	Hit@20	27.08	3.40	51.86	52.31	69.11	69.73	70.44	70.71	71.36†	72.25*
	MRR@20	17.75	1.57	17.50	21.70	29.22	29.23	30.00	31.03	31.89†	32.11*
<i>Diginetica</i>	Hit@20	21.06	5.24	26.53	35.75	46.16	49.70	45.64	47.79	50.73†	52.73*
	MRR@20	13.68	1.98	6.95	11.57	14.69	16.17	14.32	17.66†	17.59	18.05*

Table 2: Comparison of Different Models. All the results are in percentage (%). The best performing method in each column is boldfaced, and the second best method is marked with †. * indicates that the improvement of the best result is statistically significant compared with the next-best result with $p < 0.05$.

4.2 Experimental Setup

4.2.1 Evaluation Metrics. We aim to evaluate how each model performs in predicting the next item in each session. Thus, the experiments follow the leave-one-out setting with one ground-truth item to be tested for each session in the test set. As in previous works for session-based recommendation [16, 21, 32], we adopt both Mean Reciprocal Rank (MRR@K) and Hit Rate (Hit@K) as evaluation metrics. Given the ranked list of items predicted for each session, Hit@K measures the probability that the ground-truth item is within the top-K. Let r_s denote the ranking of the ground-truth item for session s . Then, $Hit_s@K = 1$ if $r_s \leq K$ and $Hit_s@K = 0$ otherwise. As for MRR@K, it measures the average ranking of the ground-truth items among the lists. That is $MRR_s@K = \frac{1}{r_s}$ if $r_s \leq K$ otherwise $MRR_s@K = 0$. Then we take the average values of MRR and Hit Rate over all the sessions in the test set and report the results.

4.2.2 Baselines.

- **S-Pop**: This simple baseline recommends the most popular items based on their popularity in the current session. Ties are broken up using the popularity values based on the whole training set.
- **KNN** [24]: This method recommends items which are most similar to items clicked in the current session. Each item is represented with a binary vector indicating all the sessions it appears in and cosine similarity is used to define their similarity.
- **MF** [22]: This matrix factorization-based model is trained with Bayesian personalized ranking loss.
- **FPMC** [23]: Building on top of both Matrix Factorization and Markov Chains, it is able to infer the next item based on sequential behaviors.
- **GRU4Rec+** [12]: This baseline utilizes Gated Recurrent Unit (GRU) to model sequential patterns, with a new ranking loss functions tailored to RNNs in session-based recommendation.

- **NARM** [16]: Besides learning the sequential behavior with GRU, it also includes an attention layer to extract the session’s main purpose, which are combined together to infer the preference scores.
- **STAMP** [19]: This model combines both general interests from long-term memory and current interests from the short-term memory of the last-clicks in session-based recommendation.
- **RPNet** [21]: Considering the repeat consumption in each session, this model can recommend unclicked items in the explore mode while recommending repeated items in the repeat mode.
- **SR-GNN** [32]: This is the state-of-the-art for session-based recommendation with graph neural networks. It models session sequences as graph-structured data and designs a graph neural network to capture the transition between items. Recommendation is made with the composition of the whole session and the last click.

4.2.3 Parameter Tuning. All of the experiments are conducted on a server machine equipped with a 12 GB Titan Xp GPU. We use Adam as the optimizer with a learning rate set to be 0.001 for all datasets. The batch size for *YooChoose*_{1/64} is 100. Since there are more training sessions in both *YooChoose*_{1/4} and *Diginetica*, we set their batch size to be 300. For a fair comparison, the dimension of the item embedding is set to be 100 as in the baseline methods. We grid search for the dropout rate in $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ and L2 regularization in $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. The dropout is set to be 0.3 and the L2 regularization is set to be 10^{-6} for all the datasets. We train each model for 50 epochs or until its performance does not improve for the validation set after 5 epochs. We fine-tune the maximum size of the sliding window in $\{2, 3, 4, 5, 6\}$ and the number of HGAT layer in $\{1, 2, 3, 4\}$ for different datasets. As in previous works, we randomly sample 10% of the training sessions as validation for parameter tuning.

4.3 Overall Evaluation. To compare SHARE with the baseline models, we summarize the overall results

Methods	<i>YooChoose</i> _{1/64}				<i>YooChoose</i> _{1/4}				<i>Diginetica</i>			
	Hit		MRR		Hit		MRR		Hit		MRR	
	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20
(1) w/o Hypergraph	59.79	70.45	29.46	30.21	60.13	70.79	29.77	30.52	37.57	50.52	16.17	17.06
(2) + GAT	60.25	70.88	29.77	30.59	60.60	71.22	30.54	31.29	38.36	51.54	17.75	17.61
(3) + HyperGCN	60.93	71.29	30.65	31.37	60.98	71.73	30.58	31.33	39.32	52.39	16.83	17.73
SHARE	61.13*	71.51*	30.67*	31.45*	61.52*	72.25*	30.96*	32.11*	39.52*	52.73*	17.12*	18.05*

Table 3: Ablation Test Result. All the results are in percentage (%). * indicates that the improvement of the best result is statistically significant compared with the next-best result with $p < 0.05$.

in Table 2. As in previous work [16, 19, 32], we report MRR and Hit Rate at $K=20$ for the overall comparison with the baselines. We find that SHARE outperforms all of the baselines under each of the metrics for session-based recommendation over all of the datasets.

Starting from the simplest method, since MF is unable to capture sequential information, it performs poorly in session-based recommendation. Integrating with Markov Chains, FPMC can improve MF, but it is still not a good fit for modeling the short-term user behaviors and performs worse than KNN. With advances in neural networks, we observe that GRU4Rec+ achieves much better results than the non-neural models in sequential pattern modeling. However, in comparison to GRU4Rec+, we see that NARM and STAMP are better suited to extract the general interests from each short-term anonymous session by aggregating items with weighted attention scores. Emphasizing the item clicked most recently helps STAMP achieve better performance than NARM. Furthermore, by predicting and modeling the repeat consumption in sessions, RpNet can outperform the attention-based and GRU-based models in this recommendation scenario.

As the state-of-the-art for session-based recommendation, SR-GNN outperforms RpNet, verifying the effectiveness of modeling item transitions with appropriate graph structure to obtain accurate user representations in sessions. However, the conventional graph structure is insufficient in capturing the correlations amongst items across different contextual windows. Since SHARE is specifically designed with this challenge in mind, we observe that it achieves significant improvements compared with SR-GNN. Further, we observe that the performance of SHARE on *YooChoose*_{1/64} is even better than SR-GNN on *YooChoose*_{1/4}. Though *YooChoose*_{1/64} and *YooChoose*_{1/4} contain different numbers of training sessions, they are tested on exactly the same set of sessions. That is, with much less training data, SHARE can recommend more accurately than the state-of-the-art model trained with more data.

4.4 Ablation Analysis. To examine the design of the proposed model, we conduct an ablation test to com-

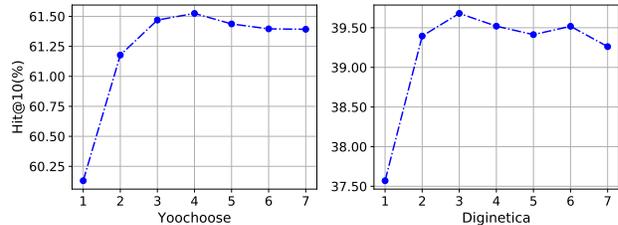


Figure 3: Size of contextual windows vs. Hit Rate.

pare SHARE with three of its variants and report the results in Table 3. In Model (1), we remove the Hypergraph attention network from SHARE, meaning that a set of static item embeddings are directly fed into the self-attention layer for different sessions. Instead of utilizing the hypergraph structure, Model (2) constructs a conventional graph for each session based on *pairwise* item co-occurrence within the session and use a Graph Attention Networks (GAT) [27] to generate the dynamic item embeddings. In Model (3), we train a hypergraph convolution networks [6] instead of HGAT network to generate the dynamic item embeddings. To compare them under different conditions, we report their performance under both $K = 10$ and $K = 20$. Overall, SHARE can outperform all of its variants for both K values, indicating the effectiveness of its design.

Without the hypergraph component, though it includes the self-attention layer to learn the sequential patterns, Model (1) achieves the weakest results since it does not consider the item correlations within each session and uses the same embedding for an item across different sessions. Then in Model (2), building on top of Model (1), we adopt the conventional graph structure and GAT networks [27] to obtain dynamic item representations considering the consecutive transitions of items in each session. This model can outperform model (1), illustrating the necessity of modeling the in-session item correlations with a graph-based structure. Next, to take advantage of the hypergraph structure for modeling the complex item correlations in each session, we construct a hypergraph for each session following the same procedure as in SHARE. But in Model (3), we generate the item embedding with hypergraph convolu-

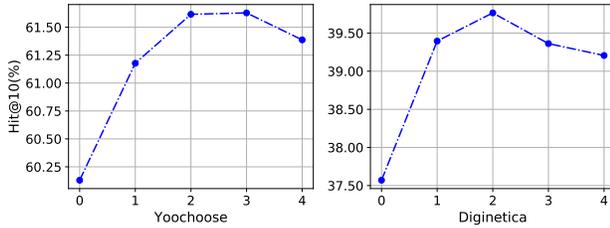


Figure 4: Number of HGAT Layers vs. Hit Rate.

tional networks [6], which aggregates information from the neighboring hyperedges with the convolutional operation. It can achieve better performance than using conventional graphs, which shows the effectiveness of the hypergraph structure and the necessity of modeling the contextual windows for item representation learning. However, it still falls behind SHARE since it does not consider the informativeness of nodes and the different impacts from contextual windows.

4.5 Study of SHARE. In this section, we conduct further studies on three core factors in SHARE: size of the contextual window, depth of information propagation and length of the sessions. In the following, due to the limit of space, as $YooChoose_{1/64}$ and $YooChoose_{1/4}$ shares the same set of testing sessions, we show the results trained with $YooChoose_{1/4}$ to uncover the patterns for YooChoose.

Size of contextual window. We first examine the performance of SHARE by varying the maximum size of contextual windows that we consider while constructing the hypergraphs (in Section 3.2). We use the 1-layer HGAT network to avoid the influence of high-order information propagation. Note that maximum size = 1 is similar to the case that no hypergraph is constructed for each session, meaning that items use static embeddings across sessions. As in Figure 3, for YooChoose, at the beginning, the performance of SHARE improves as contextual windows with larger size are incorporated into the hypergraphs. In Diginetica, the best choice of window size is smaller than in YooChoose. The reason could be that the length of sessions is shorter in Diginetica, and thus we need only consider smaller contextual windows. These observations show that contextual windows can have significant contribution on characterizing items in each session and the proposed framework is effective in modeling various contextual windows simultaneously.

High-order Propagation. We can stack multiple HGAT layers to model the information flowing among items with high-order connections in hypergraph. To visualize the high-order information propagation, while setting the maximum window size to be constant (i.e.,

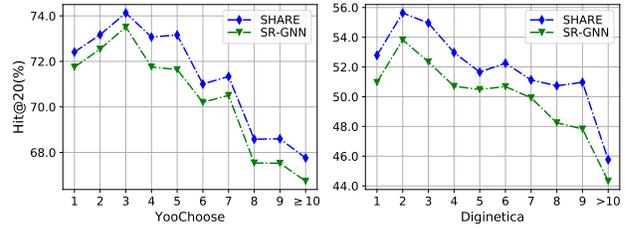


Figure 5: Session Length vs. Recommendation Quality.

2), we show the resulting Hit Rate in Figure 4 by varying the number of HGAT layers in SHARE. In each dataset, starting with a single HGAT layer, we find that the performance is improved by stacking one more layer, indicating the importance of modeling the information via high-order connections. For Diginetica, stacking more than two layers will worsen the performance since the sessions in this dataset are short in general. Using too many layers will bring in noise for the representation learning process. Meanwhile, the optimized number of layers for YooChoose is larger than that for Diginetica. Thus, we conclude that SHARE can capture both the direct and high-order connections among items in the hypergraphs and lead to accurate recommendation.

Short vs. Long Sessions. To fully understand how SHARE performs in modeling sessions under different circumstances, we group the sessions based on their length and test how SHARE and SR-GNN perform for each group of sessions. As in Figure 5, SHARE provides better recommendations for all session lengths in comparison with SR-GNN. Both models can make more accurate predictions for short action sequences than the longer ones, since they omit the order information which may be important for long sessions with adequate user behaviors. However, since the percentage of long sessions is low in real-world datasets for session-based recommendation, the main purpose is to boost the recommendation for sessions with fewer items. Indeed, the proposed framework is suitable for such practical scenarios (i.e., short-term sessions) characterized by only a limited number of sequential actions.

5 Conclusion and Future Work

To recommend the next interesting items in short-term sessions, we are motivated to exploit the correlations among items within various contextual windows in each session to better model their dynamic meanings across sessions. In this work, we propose a novel session-based recommendation system – SHARE – which is empowered by the hypergraph structure and hypergraph attention networks. With experiments on three real-world benchmark datasets, we find that the proposed SHARE is able to generate effective session-wise item embed-

dings and thus provide more accurate recommendation compared with baseline models. While in this work we mainly studying how to generate session-wise item representations, in the future we want to explore how to capture the item dynamics along time and over different domains in session-based recommendation. Meanwhile, we are also interested in extending the proposed hypergraph attention networks for representation learning in other research areas.

References

- [1] S. BAI, F. ZHANG, AND P. H. TORR, *Hypergraph convolution and hypergraph attention*, arXiv preprint arXiv:1901.08150, (2019).
- [2] R. V. D. BERG, T. N. KIPF, AND M. WELLING, *Graph convolutional matrix completion*, arXiv preprint arXiv:1706.02263, (2017).
- [3] K. DING, J. LI, R. BHANUSHALI, AND H. LIU, *Deep anomaly detection on attributed networks*, in SDM, 2019.
- [4] K. DING, J. WANG, J. LI, D. LI, AND H. LIU, *Be more with less: Hypergraph attention networks for inductive text classification*, in EMNLP, 2020.
- [5] W. FAN, Y. MA, Q. LI, Y. HE, E. ZHAO, J. TANG, AND D. YIN, *Graph neural networks for social recommendation*, in WWW, 2019.
- [6] Y. FENG, H. YOU, Z. ZHANG, R. JI, AND Y. GAO, *Hypergraph neural networks*, in AAAI, 2019.
- [7] J. GILMER, S. S. SCHOENHOLZ, P. F. RILEY, O. VINYALS, AND G. E. DAHL, *Neural message passing for quantum chemistry*, in ICML, 2017.
- [8] W. HAMILTON, Z. YING, AND J. LESKOVEC, *Inductive representation learning on large graphs*, in NeurIPS, 2017.
- [9] R. HE, W.-C. KANG, AND J. MCAULEY, *Translation-based recommendation*, in RecSys, 2017.
- [10] X. HE, Z. HE, X. DU, AND T.-S. CHUA, *Adversarial personalized ranking for recommendation*, in SIGIR, 2018.
- [11] X. HE, L. LIAO, H. ZHANG, L. NIE, X. HU, AND T.-S. CHUA, *Neural collaborative filtering*, in WWW, 2017.
- [12] B. HIDASI AND A. KARATZOGLOU, *Recurrent neural networks with top-k gains for session-based recommendations*, in CIKM, 2018.
- [13] B. HIDASI, A. KARATZOGLOU, L. BALTRUNAS, AND D. TIKK, *Session-based recommendations with recurrent neural networks*, in ICLR, 2016.
- [14] W.-C. KANG AND J. MCAULEY, *Self-attentive sequential recommendation*, in ICDM, 2018.
- [15] T. N. KIPF AND M. WELLING, *Semi-supervised classification with graph convolutional networks*, arXiv preprint arXiv:1609.02907, (2016).
- [16] J. LI, P. REN, Z. CHEN, Z. REN, T. LIAN, AND J. MA, *Neural attentive session-based recommendation*, in CIKM, 2017.
- [17] J. LI, Y. WANG, AND J. MCAULEY, *Time interval aware self-attention for sequential recommendation*, in WSDM, 2020.
- [18] G. LINDEN, B. SMITH, AND J. YORK, *Amazon.com recommendations: Item-to-item collaborative filtering*, IEEE Internet computing, 7 (2003), pp. 76–80.
- [19] Q. LIU, Y. ZENG, R. MOKHOSI, AND H. ZHANG, *Stamp: short-term attention/memory priority model for session-based recommendation*, in KDD, 2018.
- [20] R. PASRICHA AND J. MCAULEY, *Translation-based factorization machines for sequential recommendation*, in RecSys, 2018.
- [21] P. REN, Z. CHEN, J. LI, Z. REN, J. MA, AND M. DE RIJKE, *Repeatnet: A repeat aware neural recommendation machine for session-based recommendation*, in AAAI, 2019.
- [22] S. RENDLE, C. FREUDENTHALER, Z. GANTNER, AND L. SCHMIDT-THIEME, *Bpr: Bayesian personalized ranking from implicit feedback*, in UAI, 2009.
- [23] S. RENDLE, C. FREUDENTHALER, AND L. SCHMIDT-THIEME, *Factorizing personalized markov chains for next-basket recommendation*, in WWW, 2010.
- [24] B. SARWAR, G. KARYPIS, J. KONSTAN, AND J. RIEDL, *Item-based collaborative filtering recommendation algorithms*, in WWW, 2001.
- [25] Y. K. TAN, X. XU, AND Y. LIU, *Improved recurrent neural networks for session-based recommendations*, in Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, 2016.
- [26] A. VASWANI, N. SHAZEER, N. PARMAR, J. USZKOREIT, L. JONES, A. N. GOMEZ, L. KAISER, AND I. POLOSUKHIN, *Attention is all you need*, in NeurIPS, 2017.
- [27] P. VELIČKOVIĆ, G. CUCURULL, A. CASANOVA, A. ROMERO, P. LIO, AND Y. BENGIO, *Graph attention networks*, 2018.
- [28] J. WANG AND J. CAVERLEE, *Recurrent recommendation with local coherence*, in WSDM, 2019.
- [29] J. WANG, K. DING, L. HONG, H. LIU, AND J. CAVERLEE, *Next-item recommendation with sequential hypergraphs*, in SIGIR, 2020.
- [30] J. WANG, R. LOUCA, D. HU, C. CELLIER, J. CAVERLEE, AND L. HONG, *Time to shop for valentine’s day: Shopping occasions and sequential recommendation in e-commerce*, in WSDM, 2020.
- [31] X. WANG, X. HE, M. WANG, F. FENG, AND T.-S. CHUA, *Neural graph collaborative filtering*, in SIGIR, 2019.
- [32] S. WU, Y. TANG, Y. ZHU, L. WANG, X. XIE, AND T. TAN, *Session-based recommendation with graph neural networks*, in AAAI, 2019.
- [33] L. ZHENG, C.-T. LU, F. JIANG, J. ZHANG, AND P. S. YU, *Spectral collaborative filtering*, in RecSys, 2018.
- [34] Z. ZHU, J. WANG, AND J. CAVERLEE, *Improving top-k recommendation via joint collaborative autoencoders*, in WWW, 2019.