Social-aware Sparse Attention Network for Session-based Social Recommendation

Kai Ouyang[1,*], **Xianghong Xu**[1,*], **Chen Tang**[1], **Wang Chen**[2], **Hai-Tao Zheng**[1,3,†]
[1]Shezhen International Graduate School, Tsinghua Universiy
[2]Google Inc.
[3]Pengcheng Laboratory, Shenzhen, China, 518055
{oyk20, xxh20, tc20}@mails.tsinghua.edu.cn
allencw@google.com
zheng.haitao@sz.tsinghua.edu.cn

Abstract

Session-based Social Recommendation (SSR) aims to use users' social networks and historical sessions to provide more personalized recommendations for the current session. Unfortunately, existing SSR methods have two limitations. First, they do not screen users' useless social relationships and noisy irrelevant interactions. However, user preferences are mainly affected by several close friends and key interactions. Second, when modeling the current session, they do not take full advantage of user preference information. To tackle these issues, we propose a novel **S**ocial-aware **S**parse **A**ttention **N**etwork for SSR, abbreviated as **SSAN**. It mainly consists of the Heterogeneous Graph Embedding (HGE) module and the Social-aware Encoder-decoder Network (SEN) module. In the HGE module, we adopt a modified heterogeneous graph neural network, which focuses more on close friends and key historical interactions, to enhance user/item representations. In the SEN module, we use the user representation as a bridge between the Encoder and Decoder to incorporate user preferences when modeling the current session. Extensive experiments on two benchmark datasets demonstrate the superiority of SSAN over the state-of-the-art models.

1 Introduction

Session-based Social Recommendation (SSR) is proposed based on Session-based Recommendation (SR). Initially, SR aims to predict the next item for the current anonymous session. The anonymous session is a sequence of items clicked in a transaction, without user IDs and users' social networks. Later, with the boom of modern network and social media, user IDs can be tracked, and it becomes more common for users to have a social network. Therefore, SSR has been proposed, which aims to capture user preferences based on their social networks and historical sessions to provide more personalized recommendations (Song et al., 2019).

SSR is initially proposed in DGRec (Song et al., 2019), which uses graph neural networks (GNNs) to aggregate the preferences of neighbors for each user. Afterward, SERec (Chen and Wong, 2021) proposes to use a heterogeneous graph neural network to learn user and item representations that integrate the knowledge from social networks. Since they take advantage of users' social networks and historical interactions, they have a significant performance improvement over previous conventional SR methods (Li et al., 2017; Liu et al., 2018).

However, most of the existing SSR models suffer from two defects: **(a)** They follow a strong underlying assumption that users' all friends and interactions can influence their preferences. Therefore, they aggregate the features of all friends and historical interactions to model the user and do not screen the irrelevant items when modeling the current session. However, user preferences are mainly affected by several close friends and key interactions. Moreover, it has been shown in the literature that irrelevant items can interfere with session modeling (Yuan et al., 2021). In other words, aggregating all information without filtering can lead to bias in modeling user preferences. **(b)** When modeling the current session, they do not make full use of the user's personalized information. For example, SERec (Chen and Wong, 2021) only concatenates user representations at the last stage of model inference, which limits the expressiveness of the personalized knowledge of the user.

To tackle these issues, we propose to eliminate the low-confidence information, and incorporate personalized knowledge into the modeling of the current session. Hence, we put forward a novel **S**ocial-aware **S**parse **A**ttention **N**etwork for SSR, abbreviated as **SSAN**. It mainly consists of the Heterogeneous Graph Embedding (HGE) module and

[*] Equal contribution.
[†] Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 2173–2183
December 7-11, 2022 ©2022 Association for Computational Linguistics

the Social-aware Encoder-decoder Network (SEN) module. The HGE module aims to model user preferences based on users' social relationships and historical interactions. In the HGE module, we use a heterogeneous graph neural network, which focuses more on close friends and important historical interactions, to enhance user/item representations. It can alleviate the impact of invalid social relationships and useless historical interactions on user modeling. The SEN module aims to model the current session based on user preferences and interactions in the current session. In the SEN module, we mine the latent intents from the interactions and inject user preference information into the modeling process of the current session. It can alleviate the impact of unreliable interactions and take full advantage of the personalized knowledge of users. Extensive experiments on two public benchmark datasets demonstrate the superiority of **SSAN**. Further ablation experiments demonstrate the effectiveness of the HGE and SEN modules.

To summarize, we mainly make the following contributions:

- **Mine latent key information.** We construct the HGE module that concentrates on close friends and key historical interactions to enhance user/item representations. Besides, we use the sparse transformation function to mitigate the impact of irrelevant interaction items.
- **Integrate personalized knowledge.** We devise the SEN module to closely integrate user preference information to make more personalized recommendations for current session.
- **Excellent performance.** We perform extensive comparisons with recent SSR and SR methods on two public real-world datasets, demonstrating the superiority of SSAN.

2 Related Work

In this section, we retrospect the existing work related to our research, which mainly consists of the following three subsections.

2.1 Session-based Recommendation

Session-based Recommendation can be mainly divided into Anonymous Session-based Recommendation (ASR) and Personalized Session-based Recommendation (PSR).

2.1.1 Anonymous Session-based Recommendation (ASR)

Let $I = \{i_1, i_2, \dots, i_N\}$ denote the set of items, where N is the total number of items. A session is represented as a list $S = [i_{s,1}, i_{s,2}, \dots, i_{s,t}]$ ordered by the timestamp and $i_{s,k} \in I (1 \leq k \leq t)$ represents an interacted item of the anonymous user. The task of ASR is to predict the next item $i_{s,t+1}$ for an anonymous session S.

Early ASR studies (Rendle et al., 2010) focused on extracting sequence information from session data using Markov chains. Following these works, GRU4Rec (Hidasi et al., 2015) is the first research that formally defines ASR and proposes a multi-layered GRU model. NextItNet (Yuan et al., 2019) applies dilated convolutional layers to model the local item dependence. Recently, GNNs have drawn increasing attention in various tasks, including ASR. SR-GNN (Wu et al., 2019) represents sessions as directed subgraphs and apply GNN to capture the item transitions. GCE-GNN (Wang et al., 2020) exploits global-level item-transitions over all sessions to learn global-level contextual information. Since these methods are designed for the anonymous session, they do not leverage the knowledge of users' social networks. Moreover, most of them ignore the randomness of user behavior and do not consider the reliability of user interactions.

2.1.2 Personalized Session-based Recommendation (PSR)

Let the sets of users and items be denoted by $U = \{u_1, u_2, \dots, u_M\}$ and $I = \{i_1, i_2, \dots, i_N\}$, respectively. The historical session set $\mathcal{D}$ contains all sessions of each user. Let $\mathcal{D}^u = \{S_1^u, S_2^u, \dots, S_{|\mathcal{D}^u|}^u\}$ represents the session set associated with user $u \in U$, where $S_T^u \in \mathcal{D}^u$ denotes the T^{th} session of user u, and $S_T^u[t] \in I$ denotes the t^{th} item in session S_T^u. The task of PSR is to predict the next item $S_T^u[t+1]$ for session S_T^u. Different from ASR, PSR knows which user the sequence belongs to, so it can model the user's preferences and exploit them.

In recent years, various attempts have been made for PSR. Quadrana et al. (Quadrana et al., 2017) use hierarchical recurrent neural networks to capture users' evolving interests. Then, Zhang et al. (Zhang et al., 2020) explicitly model the effect of the users' historical interests on the current session by the attention mechanism. Guo et al. (Guo et al., 2019) improve the attention mechanism by applying Matrix Factorization to users' historical interactions. These methods leverage users' long-term interaction history to provide more personalized recommendations, but they fail to capture the impact of users' social networks.

2.2 Social Recommendation

It is a growing trend towards leveraging social networks to make recommendations more personalized and effective. Ma et al. (Ma et al., 2011) regularize the latent user factors so that connect users with similar latent factors and make recommendations. Zhao et al. (Zhao et al., 2014) apply matrix factorization to extract additional training instances from social networks. Wang et al. (Wang et al., 2017) propose to distinguish strong and weak relationships and learn personalized preferences from social networks. Xiao et al. (Xiao et al., 2017) propose to model user-items interactions and recognize the social relationships of the user using transfer learning. Wang et al. (Wang et al., 2019) maintain a heterogeneous social graph to extract the social knowledge to enhance the user representations. These methods only utilize collaborative information from user-item interactions and users' social networks without considering the sequential information of interactions. Thus, they are not suitable for the session-based recommendation.

2.3 Session-based Social Recommendation (SSR)

SSR is proposed to predict users' next click in the current short-term session based on social networks and historical sessions. It aims to combine the advantages of session-based recommendation and social recommendation and provide more accurate and personalized recommendations. The first SSR model is DGRec (Song et al., 2019), which uses a graph attention network to model the social influence of the user. Then, SERec (Chen and Wong, 2021) proposes to use a heterogeneous graph to process related users and items when making predictions for the current session.

Unfortunately, while these models are laudable attempts to integrate social networks for the session-based recommendation, they still fail to take full advantage of the personalized knowledge of the user when modeling the current session. Moreover, they ignore the fact that users' preferences are influenced mainly by several close friends and key interactions.

3 Task Definition

Let $U = \{u_1, u_2, \ldots, u_M\}, I = \{i_1, i_2, ..., i_N\}$ denote the set of users and items, respectively. M, N are the total number of users and items, respectively. Let $\mathcal{D}$ represents the set of all historical sessions of users. The set of all sessions of a user is represented by $\mathcal{D}^u = \{S_1^u, S_2^u, \ldots, S_{|\mathcal{D}^u|}^u\}$, where session $S_T^u = [i_1^u, i_2^u, ..., i_t^u]$ is the T^{th} session in $\mathcal{D}^u$ containing a sequence of interacted items of user u. We denote by $C_T^u = [c_1^u, c_2^u, ..., c_t^u]$ the original embedding set corresponding to S_T^u. For briefly, the superscript and/or the subscript in S_T^u and C_T^u may be dropped if there is no ambiguity. Different from PSR, SSR has a social network for each user, which is a graph denoted as $\mathcal{G} = (U, E)$. The node set U is the user set U, and the edge set E indicates the users' social relationships. Specifically, an edge $(u, v) \in E$ from user u to user v represents u is followed by v.

The task of SSR is to predict the next item of a new session $S^u \notin \mathcal{D}^u$ based on social network $\mathcal{G}$ and the set of all previous sessions $\mathcal{D}^u$. It can be formalized as predicting the probability of user interaction with each item $i \in I$ at time step $t + 1$:

$$p\left(i_{t+1}{}^{(S^u)} = \hat{i} \mid (S^u, \mathcal{G}, \mathcal{D}^u)\right), \tag{1}$$

where $\hat{i} \in I$ represents the candidate item.

Since a recommender usually needs to provide multiple recommendations for users, SSR will recommend top-K items according to the scores.

4 Method

In this section, we introduce the SSAN in detail.

4.1 Sparse Transformation Function

In general, the attention mechanism uses softmax (Bridle, 1990) to convert weights into probabilities. Essentially, softmax is a mappings function from $\mathbb{R}^d$ to $\triangle^{d-1}$, where $\triangle^{d-1} = \{p \in \mathbb{R}^d | 1^\top p = 1, p \geq 0\}$. However, it may assign weights to the useless data due to its nonzero probability, affecting the ability to find the relevant items. Then, a sparse transformation method is proposed to assign zero for the low-scoring items, named sparsemax (Martins and Astudillo, 2016):

$$\text{sparsemax}(x) = \underset{p \in \triangle^{d-1}}{\arg\max} \|p - x\|^2 \tag{2}$$

where x is the input weights, and p is the output probabilities vector. Recently, a novel transfor-

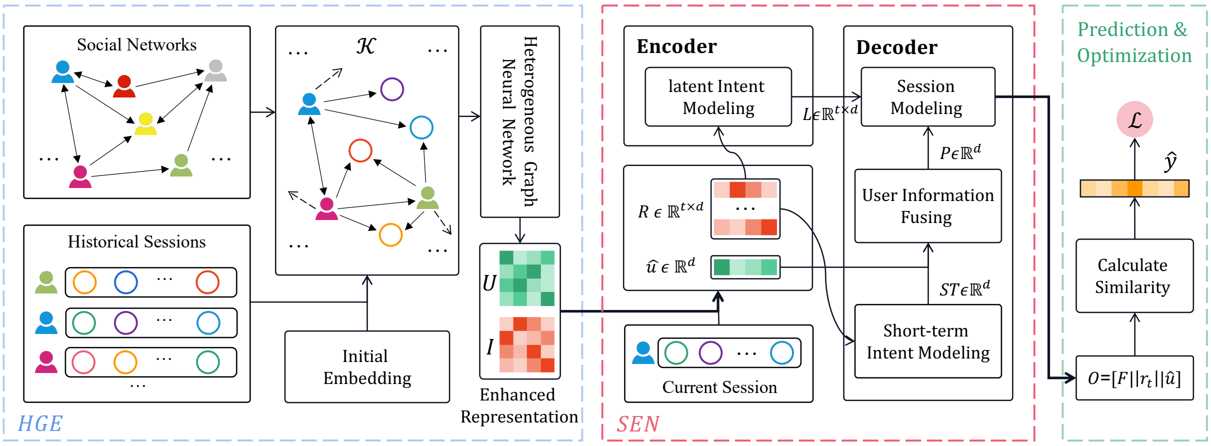

Figure 1: The architecture of SSAN. It is mainly composed of HGE and SEN modules.

mation function α-entmax (Peters et al., 2019) is proposed to replace the softmax:

$$\alpha\text{-entmax}(x) = \arg\max_{p\in\triangle^{d-1}} p^{\top}x + H_{\alpha}^{\top}(p), \text{where}$$

$$H_{\alpha}^{\top}(p) = \begin{cases} \frac{1}{\alpha(\alpha-1)}\sum_{j}(p_j - p_j^{\alpha}) & ,\alpha \neq 1 \\ H^{S}(p) & ,\alpha = 1 \end{cases} \tag{3}$$

where $H_{\alpha}^{\top}(p)$ is the Tsallis α-entropies (Tsallis, 1988). In particular, 1-entmax equals the softmax function and 2-entmax (Peters et al., 2019) equals the Sparsemax. In this paper, we replace the transformation function with α-entmax in the attention mechanism, to filter the irrelevant interactions in the current session.

4.2 Architecture

The architecture of SSAN is depicted in Figure 1. SSAN mainly consists of the following parts:

- **Heterogeneous Graph Embedding (HGE) module.** In this module, we employ social networks and historical sessions to enhance the user/item representations.

- **Social-aware Encoder-decoder Network (SEN) module.** In this module, we integrate user preference information based on the sparse attention network to model the current session. Besides, we also mine the short-term intent of the user.

- **Prediction and Optimization.** In this part, we evaluate the probability of each candidate item using the final session representation.

We will introduce the above parts in detail in the rest of this section.

4.3 Heterogeneous Graph Embedding (HGE) module

To integrate users' social networks and historical sessions to enhance user/item representations, we construct the HGE module inspired by SERec (Chen and Wong, 2021). Moreover, it can alleviate the impact of useless social relationships and invalid interactions when modeling user preferences.

4.3.1 Build Heterogeneous Graph

In this layer, we build a heterogeneous graph based on users' social networks and historical sessions.

Formally, let $\mathcal{K} = \{\mathcal{N}, \mathcal{E}\}$ be the heterogeneous graph. For simplicity, we use the symbols of the users U and items I to indicate the type of the corresponding node. $\mathcal{N} = U \cup I$ denotes the node set of the graph consisting of all users and items involved in $\mathcal{D}$. $\mathcal{E}$ is the edge set contains three types of directed edges, *i.e.*, *user-user* edges (UU), *user-item* edges (UI),and *item-item* edges (II).

Specifically, a *user-user* edge $(u, v) \in \mathcal{E}$ if user u is followed by user v. a *user-item* edge $(u, i) \in \mathcal{E}$ if user u clicks item i in any session. Besides, a *user-item* edge $(i_1, i_2) \in \mathcal{E}$ if a transition from i_1 to i_2 appears in any session.

4.3.2 Learn Enhanced Representation

To capture the user preference information contained in the user's social relationships and historical interactions, we apply the heterogeneous graph neural network on the graph $\mathcal{K}$.

Let $\mathcal{R}^{l}[v]$ denotes the representation of node v at layer l, where v is a user or an item, where $\mathcal{R}^{0}[v] \in \mathbb{R}^{d}$ is the initial user/item embedding, where d is the embedding size. To generate a new representation for node v from layer $l-1$ to layer

l, we calculate the importance of each connection node of v in the graph $\mathcal{K}$:

$$a_{uv} = \theta^l \cdot (\sigma(\mathrm{MLP}(\mathcal{R}^{l-1}[u] \cdot \mathcal{R}^{l-1}[v])) + e^l), \tag{4}$$

where $\theta^l \in \mathbb{R}^d$ is a learnable parameter, $e^l \in \mathbb{R}^d$ is the feature vector of edge (u, v), $u, v \in \mathcal{N}$, σ denotes the sigmoid activation function, MLP indicates the multi-layer perceptron, $\mathcal{R}^{l-1}[u]$ means the representation of node u at layer $l-1$, and $\cdot$ means the element-wise multiplication. Then, we normalize the score a_{uv}:

$$\hat{a}_{uv} = \frac{a_{uv}}{\sum_{k \in \mathcal{H}_u} a_{uk}}, \tag{5}$$

where $\mathcal{H}_u$ indicates the neighbors of node u. We argue that the neighbors with low weight may not benefit to the update of the representation, so we set the values $\hat{a}_{uv}$ less than β as 0:

$$\widetilde{a}_{uv} = \begin{cases} \hat{a}_{uv}, & \hat{a}_{uv} > \beta. \\ 0, & \hat{a}_{uv} \le \beta. \end{cases} \tag{6}$$

Then, we re normalize them so that $\sum \widetilde{a}_{uv} = 1$. Finally, we aggregation the neighbor nodes by:

$$\begin{aligned} \widetilde{\mathcal{R}}^{l-1}[u] &= \sum_{k \in \mathcal{H}_u} \widetilde{a}_{uk}(\mathrm{MLP}(\mathcal{R}^{l-1}[k])), \\ \mathcal{R}^l[u] &= \mathrm{ReLU}(\mathrm{MLP}([\widetilde{\mathcal{R}}^{l-1}[u] || \mathcal{R}^{l-1}[u]])), \end{aligned} \tag{7}$$

where ReLU is the activation function, and || denotes concatenate operation.

In particular, in the HGE module, we set different MLP for different edges and different layers. After L_{gnn} layers of the above process, we obtain $\mathcal{R} = \mathcal{R}^L[\mathcal{N}]$, which is the final enhanced user/item representation set.

4.4 Social-aware Encoder-decoder Network (SEN) module

To make full use of user preference information and alleviate the negative impact of users' unreliable interaction signals, we construct the SEN module. It is mainly made up of the Encoder and Decoder.

4.4.1 Encoder

The Encoder aims to mine the user's latent intent sequence based on the interaction sequence of the current session.

First, for a session $S = [i_1, i_2, \ldots, i_t]$ corresponding to a user u, where t is the length of current session, we can obtain the enhanced item representation set $R = [r_1, r_2, \ldots, r_t]$, $r \in \mathbb{R}^d$, and the enhanced user representation $\hat{u} \in \mathbb{R}^d$ from set $\mathcal{R}$. To capture the sequence information of the session, we employ a learnable positional embedding module (Sun et al., 2019). Formally, for each item i of input session, the hidden representation is:

$$x = r + p, \tag{8}$$

where $x \in \mathbb{R}^d$ denotes the hidden representation of item i, and $p \in \mathbb{R}^d$ is the position embedding. Thus, we obtain the hidden representation set $X = [x_1, x_2, \ldots, x_t] \in \mathbb{R}^{t \times d}$ for session S.

Latent Intent Modeling. In this layer, we mine the latent intents of the user based on the interaction sequence of the current session and eliminate the low-confidence items. Specifically, we use the sparse attention network to encode the interaction sequence to obtain the latent intent sequence:

$$\begin{aligned} Q &= \mathrm{ReLU}(\mathrm{MLP}(X)), \\ L' &= \mathrm{SparseAttention}(Q, X, X), \end{aligned} \tag{9}$$

where $L' \in \mathbb{R}^{t \times d}$, MLP indicates the multi-layer perceptron, ReLU is the activation function, and the SparseAttention can be formalized as:

$$\mathrm{SparseAttention}(Q, K, V) = \alpha\text{-entmax}(\frac{Q^\top K}{\sqrt{d}})V, \tag{10}$$

where Q, K, and V are the input matrices, and SparseAttention is a multi-head network. It is worth mentioning that the model performance is not sensitive to the number of heads, so we empirically set it to 4. Besides, we use the mask matrix to ensure that the mining for t-th item can depend only on its previous items. Then we endow the model with more non-linearity:

$$L' = \mathrm{ReLU}(\mathrm{MLP}(L')), \tag{11}$$

where MLP indicates the multi-layer perceptron, and all sessions will share the same parameters.

After that, we add a Residual Connection and Layer Normalization to the result to alleviate the instability of model training. We also add the dropout mechanism to alleviate the overfitting. The Encoder is stackable, and we let L_{enc} denote the number of the Encoder layers. The output of last layer is $L = [l_1, l_2, \ldots, l_t] \in \mathbb{R}^{t \times d}$, which represents the latent intent sequence.

4.4.2 Decoder

The Decoder aims to achieve the social-aware modeling of the current session based on user preference information and the latent intent sequence.

Short-term Intent Modeling. To mine more sequence information in the interaction sequence, we employ GRU (Cho et al., 2014) on the enhanced interaction sequence:

$$ST = \text{GRU}(R), \tag{12}$$

where $R \in \mathbb{R}^{t \times d}$ is the set of enhanced item representation. $ST \in \mathbb{R}^d$ is the last hidden state of GRU, and we view it as the short-term intent representation of the current session.

User Information Fusing. Considering that only focusing on the interactions in the current session and ignoring the use of user preference information will limit the performance of the model. In this layer, we integrate the user preference information into the modeling of the current session.

Specifically, we fuse the representation of short-term intent ST with the enhanced user representation $\hat{u}$ from the HGE module, to obtain the personalized intent representation:

$$P = \text{ReLU}(\text{MLP}(ST + \hat{u})), \tag{13}$$

where $P \in \mathbb{R}^d$ is the personalized intent representation that integrates user preference information and short-term intent.

Session Modeling. In this layer, we implement the social-aware decoding on the output L of the Encoder based on the personalized representation. Technically, we input P and the latent intent sequence L into the sparse attention network for decoding to model the current session:

$$\begin{aligned} F' &= \text{SparseAttention}(P, L, L), \\ F' &= \text{ReLU}(\text{MLP}(F')), \end{aligned} \tag{14}$$

where $L \in \mathbb{R}^{t \times d}$ is the final output of the Encoder.

Similarly, we add a Residual Connection and Layer Normalization on the result to alleviate the instability of model training, and also add the dropout mechanism to alleviate the overfitting. Moreover, the Decoder is stackable, and we let L_{dec} denote the number of the Dncoder layers. The output of the last layer is $F \in \mathbb{R}^d$, which is the final session representation.

4.5 Prediction and Optimization

In this part, we complete the prediction of the current session. First, to capture the user's intent at the end of the session and make full use of the user's preference information, we integrate some key information:

$$O = W_o[F||r_t||\hat{u}], \tag{15}$$

where $O \in \mathbb{R}^d$ is the final representation used to make recommendations, $r_t \in \mathbb{R}^d$ represents the enhanced representation of the last item, $\hat{u} \in \mathbb{R}^d$ is the enhanced user representation, and $W_o \in \mathbb{R}^{d \times 3d}$ is the projection matrix.

Since the next item prediction can convert into a probability distribution of items, we calculate the similarity of all items to the representation O:

$$\mathbf{z}_i = O^\top c_i, \tag{16}$$

where $c_i \in \mathbb{R}^d$ is the initial embedding of candidate item $i \in I$, and $\mathbf{z}_i$ is similarity score. We use the softmax function to normalize the similarity score:

$$\hat{\mathbf{y}}_i = \text{softmax}(\mathbf{z}_i), \tag{17}$$

where $\hat{\mathbf{y}}_i$ is the probability of item i appearing in the next click in the current session.

For any given session, the loss function is defined as the cross-entropy of the ground truth $\mathbf{y}_i$ and the prediction result $\hat{\mathbf{y}}_i$:

$$\mathcal{L} = -\sum_{i=1}^{I} \mathbf{y}_i \log(\hat{\mathbf{y}}_i) + (1 - \mathbf{y}_i) \log(1 - \hat{\mathbf{y}}_i), \tag{18}$$

where $\mathbf{y}$ is the ground truth probability distribution, which is a one-hot vector.

5 Experiments

We conduct experiments on two real-world benchmark datasets and mainly aim to answer the following research questions:

RQ1: How does **SSAN** compare to other state-of-the-art (SOTA) models?

RQ2: Whether using the users' social networks and historical sessions is conducive to predicting the user's next click?

RQ3: Whether **HGE** module is beneficial to the final performance?

RQ4: Whether **SSAN** is efficient?

RQ5: How the modules and layers of **SSAN** affect the final performance?

5.1 Experimental Settings

Dataset	# items	# sessions	# users
Gowalla	41,229	258,732	33,661
Delicious	5793	60,397	1313

Table 1: Statistics of the two datasets.

5.1.1 Datasets

We conduct extensive experiments on the following two public datasets:

Gowalla[1]: it comes from a location-based social networking website, where users can share their location by checking in. Following SERec (Chen and Wong, 2021), we divide the two check-in records into two sessions, if the interval between them is longer than 1 day. **Delicious**[2]: it comes from an online bookmarking system, where users can assign various semantic tags to bookmarks. Following SERec, we take a series of tag operations with a small timestamp gap as a session.

We follow the same data processing method as SERec. For each dataset, we take the first 60% as the training set, 20% as the validation set, and the rest 20% as the test set. Then, we filter the short sessions and the infrequent items, and apply a data augmentation technique described in SERec on these two datasets. The statistics of datasets after preprocessing are shown in Table 1.

5.1.2 Evaluation Metrics

We adopt three commonly used ranking-based metrics to evaluate all models: HR@K (Hit Ratio), MRR@K (Mean Reciprocal Rank (Voorhees, 2001)), and NDCG@K (Normalized Discounted Cumulative Gain (Järvelin and Kekäläinen, 2000)), where the values of K included $\{10, 20\}$.

5.1.3 Implementation Details

For a fair comparison, we implement our model on the public pre-processed version datasets provided by SERec (Chen and Wong, 2021). Not only that, we also following SERec to make the following settings: We use Adam (Kingma and Ba, 2015) optimizer with learning rate 0.001, and the weight decay coefficient is 0.0001. We use 128-dimensional embeddings for items and users. We apply early stopping if the performance does not improve in 2 epochs on the validation set. We set the number of epochs to 30 and set the mini-batch size to 128.

Besides, we search for the number of GNN layers L_{gnn} in $\{1, 2, 3\}$ and finally set it to 1. The L_{enc} and L_{dec} are tuned amongst $\{1, \cdots, 4\}$ and finally set to 3 and 1, respectively. The α of α-entmax is tuned amongst $\{1.1, \cdots, 1.9\}$ and finally set to 1.5. The β in HGE module is tuned amongst $\{0.01, 0.02, \cdots, 0.2\}$ and finally set to 0.05.

5.1.4 Baselines

We compare SSAN with the following representative SOTA recommendation methods. They can be categorized into session-based recommendation models and SSR models:

Item-KNN (Sarwar et al., 2001), which recommends items similar to the last item in the session. **FPMC** (Rendle et al., 2010), which is a traditional sequential method based on Markov Chain. **NARM** (Li et al., 2017) [3], which utilizes RNN and attention mechanism to capture the main purpose of the session. **STAMP** (Liu et al., 2018) [4], which uses the self-attention mechanism to captures the long-term and short-term preferences of sessions. **SR-GNN** (Wu et al., 2019) [5], which employs gated graph convolutional neural networks to capture complex transitions of items to achieve promising results. **SSRM** (Guo et al., 2019), which proposes a Matrix Factorization based attention model. **NextItNet** (Yuan et al., 2019) [6], which is a classic CNN-based method for sequential recommendation. **GCE-GNN**[7] (Wang et al., 2020), which is a widely compared GNN-based model that learns global and local information of sessions. **DSAN** (Yuan et al., 2021) [8], which utilizes sparse attention mechanism to alleviate the effect of unrelated items that clicked by users.

And we compare SSAN with the following SSR models: **DGRec** (Song et al., 2019) [9], which uses RNN and graph attention neural network to model the dynamic interests and social influences. **SERec** (Chen and Wong, 2021) [10] is the state-of-the-art method for SSR, which has an efficient and effective knowledge embedding framework.

For a fair comparison, our implementation provides user and item representations of the Heterogeneous Graph Embedding (HGE) module for SR models. In addition, since some models miss some metrics to varying degrees in the public results, for a fair comparison, we report the best results in the original paper (if available) and the reproduced results on the same device as SSAN.

[1] https://snap.stanford.edu/data/loc-gowalla.html

[2] https://grouplens.org/datasets/hetrec-2011/

[3] https://github.com/lijingsdu/sessionRec_NARM

[4] https://github.com/uestcnlp/STAMP

[5] https://github.com/CRIPAC-DIG/SR-GNN

[6] https://github.com/fajieyuan/WSDM2019-nextitnet

[7] https://github.com/CCIIPLab/GCE-GNN

[8] https://github.com/SamHaoYuan/DSANForAAAI2021

[9] https://github.com/chensi01/DGRec

[10] https://github.com/twchen/SEFrame

Dataset	Gowalla						Delicious					
Model	**H10**	**M10**	**N10**	**H20**	**M20**	**N20**	**H10**	**M10**	**N10**	**H20**	**M20**	**N20**
ItemKNN	33.27	18.47	-	39.11	18.88	-	20.84	9.98	-	27.82	10.46	-
FPMC	35.31	17.66	-	42.57	18.17	-	29.59	14.46	-	38.26	15.02	-
NARM	41.56	22.50	26.23	49.55	23.04	28.24	37.18	19.76	23.88	46.39	20.40	26.23
STAMP	41.93	22.55	25.48	49.68	23.10	27.47	37.56	19.95	24.10	46.48	20.57	26.37
NextItNet	39.87	21.51	25.62	47.80	22.12	26.92	35.14	18.04	22.53	44.62	18.69	25.43
SR-GNN	41.56	22.39	26.88	49.58	22.94	28.94	37.98	20.37	24.65	47.41	20.99	26.92
SSRM	41.63	22.45	26.88	49.64	22.98	28.87	37.51	19.83	23.83	46.57	20.46	26.15
GCE-GNN	41.88	22.56	26.93	49.77	23.02	28.96	37.32	19.82	23.67	46.40	20.36	26.14
DSAN	41.75	22.71	27.23	49.60	23.24	29.18	37.54	19.96	24.09	46.53	20.62	26.38
$\mathcal{G}$_NARM	44.94	24.03	29.16	52.64	24.56	31.11	39.82	21.09	25.46	49.15	21.73	27.96
$\mathcal{G}$_STAMP	45.42	24.44	29.59	52.88	25.16	31.49	38.93	20.66	24.94	48.45	21.32	27.18
$\mathcal{G}$_NextItNet	45.83	25.03	29.92	53.39	25.55	31.79	39.32	21.02	25.47	48.63	21.67	27.78
$\mathcal{G}$_SR-GNN	45.45	25.04	29.96	53.15	25.57	31.87	39.96	21.37	25.76	49.15	22.01	28.08
$\mathcal{G}$_SSRM	45.51	24.44	29.46	53.27	24.98	31.42	40.02	21.24	25.68	49.43	21.90	28.06
$\mathcal{G}$_GCE-GNN	45.90	25.09	29.95	53.54	25.98	31.71	40.12	21.19	25.71	49.32	21.35	27.95
$\mathcal{G}$_DSAN	46.00	25.39	30.11	53.63	25.93	31.64	40.01	21.26	25.98	49.54	21.40	28.06
DGRec	42.10	23.11	27.49	49.98	23.64	29.51	37.60	20.22	24.33	47.19	20.86	26.64
SERec	46.01	25.14	30.10	53.72	25.67	31.96	40.02	21.29	25.88	49.53	21.98	28.15
SSAN	**46.67**	**25.95**	**30.90**	**54.61**	**26.51**	**32.91**	**42.66**	**23.52**	**28.07**	**52.27**	**24.15**	**30.39**
Improv. (%)	1.43	2.21	2.62	1.66	2.00	2.97	6.33	10.06	8.04	5.51	9.72	7.96

Table 2: The performance of all models in % on Gowalla and Delicious datasets, where **H**, **M**, and **N** represent HR, MRR, and NDCG, respectively. $\mathcal{G}$ represents that the model uses **HGE** module, *i.e.*, the model uses the enhanced user/item representations. The best result is in bold, and the second best result is underlined.

5.2 Experimental Results

In this section, we investigate SSAN in detail according to the experimental results.

5.2.1 Overall Performance

The experimental results of overall performance are reported in Table 2, and we can draw the following conclusions:

(RQ2). All variants of non-social-aware methods (e.g., $\mathcal{G}$_NARM) significantly outperform original models (e.g., NARM), which strongly demonstrates the superiority of using social knowledge.

(RQ3). The performance of these variants is close to or even better than SERec, which shows the superiority of the HGE module which pays attention to close friends and important interactions.

(RQ1). SSAN is overwhelmingly superior to all baseline models, which indicates the effectiveness of SSAN. We believe the performance improvement of our model mainly comes from the following aspects: (1) We construct the HGE module, which uses an improved heterogeneous graphical neural network to inject social knowledge and historical interaction information into the user modeling process. (2) We construct the SEN module based on the sparse transformation function. It can capture the sequence information and fully integrate the user preference information during decoding. In a word, our efforts in more effectively using the knowledge of social networks and filtering out irrelevant items make us achieve better performance.

Model	Training	Inference	Model	Training	Inference
NARM	9.62	6.42	$\mathcal{G}$_NARM	48.29	6.68
STAMP	9.31	5.74	$\mathcal{G}$_STAMP	45.56	5.96
SSRM	12.93	6.76	$\mathcal{G}$_SSRM	46.98	6.95
SR-GNN	69.22	68.32	$\mathcal{G}$_SR-GNN	93.24	69.58
DSAN	48.23	27.48	$\mathcal{G}$_DSAN	90.30	31.98
SERec	99.44	78.26	**SSAN**	98.96	35.13

Table 3: Running time in seconds (s) per 1000 batches.

5.3 Efficiency of SSAN (RQ4)

To investigate the efficiency of **SSAN**, we compared some models' running time during both training and inference on the Delicious dataset. The experimental results are shown in Table 3. We can observe that those variants (e.g., $\mathcal{G}$_NARM) of non-social-aware methods run slightly slower than original models (e.g., NARM) during training and run as fast as their original models during inference. In the real world, as long as the model can have better performance, a lightly large time cost of training is acceptable, since the model only needs to be

Dataset	Delicious					
Model	H10	M10	N10	H20	M20	N20
SSAN-noHGE	42.07	22.77	27.34	51.32	23.41	29.68
SSAN-noSEN	41.47	22.62	27.08	51.34	23.29	29.54
SSAN-noST	40.07	21.54	25.92	49.62	22.20	28.33
SSAN-noU	42.16	23.21	27.79	51.65	23.75	30.08
SSAN	**42.66**	**23.52**	**28.07**	**52.27**	**24.15**	**30.39**

Table 4: Performance of the variants of SSAN.

trained once in a period. Although the HGE module which captures social knowledge requires more training time, it does not need additional time in the inference process, which demonstrates that using social knowledge is feasible. SSAN achieves significant performance improvement without requiring more training time than SERec, which shows the efficiency and superiority of SSAN.

5.4 Ablation Study (RQ5)

To investigate different modules and explore the effectiveness of some layers in SSAN, we compared four variants of SSAN with the original SSAN. According to the experimental results shown in Table 4, we can draw the following conclusions:

(1) **SSAN-noHGE** denotes SSAN without the HGE module. **SSAN-noSEN** represents SSAN without the SEN module, and we make prediction using the mean of all item embedding in the session. These two variants have a pretty poor performance, which indicates HGE and SEN modules are beneficial to the final performance, and also shows the effectiveness of the two modules.

(2) **SSAN-noST** means SSAN without **Short-term Intent Modeling** layer. We can observe that its performance degrades significantly, which indicates the step of modeling the short-term intent is essential for SSAN.

(3) **SSAN-noU** denotes SSAN without **User Information Fusing** layer. We can observe that its performance has also declined, which indicates that it is important to use user preference information to predict the next item of the current session.

6 Conclusion

In this paper, we summarize two issues of previous SSR methods. To tackle these issues, we propose a novel Social-aware Sparse Attention Network, abbreviated as SSAN. In this model, we construct the HGE module based on the improved heterogeneous graph neural network. It can inject the high-confidence social knowledge and historical interaction information into the modeling process of users and items. Meanwhile, we construct the SEN module based on the sparse attention mechanism to integrate user preference information when modeling the current session. Extensive experimental results on two datasets demonstrate the superiority of SSAN over the state-of-the-art models. In future work, we plan to explore more efficient methods to capture the social knowledge and enhance the ability in screening irrelevant items.

7 Limitations

In this section, we discuss the limitations of our work in detail and propose corresponding solutions that we believe are feasible. SSAN aims to leverage users' social networks and historical sessions to provide more personalized recommendations for the current session. We argue that its limitations mainly lie in the rough confidence judgment method, and the slightly higher time cost.

(1) Our screening method for useless social relationships and invalid historical interactions is simple. We believe that we can introduce the idea of contrastive learning to construct a more reliable confidence judgment method.

(2) We have a slightly higher training cost. We believe we can eliminate the layers with a lower cost–performance ratio by conducting more ablation experiments to achieve the trade-off between time cost and performance.

Acknowledgements

This research is supported by National Natural Science Foundation of China (Grant No.62276154 and 62011540405), Beijing Academy of Artificial Intelligence (BAAI), the Natural Science Foundation of Guangdong Province (Grant No. 2021A1515012640), Basic Research Fund of Shenzhen City (Grant No. JCYJ20210324120012033 and JSGG20210802154402007), and Overseas Cooperation Research Fund of Tsinghua Shenzhen International Graduate School (Grant No. HW2021008).

References

John S Bridle. 1990. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer.

Tianwen Chen and Raymond Chi-Wing Wong. 2021. An efficient and effective framework for session-based social recommendation. In *Proceedings of the*

14th ACM International Conference on Web Search and Data Mining, pages 400–408.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Lei Guo, Hongzhi Yin, Qinyong Wang, Tong Chen, Alexander Zhou, and Nguyen Quoc Viet Hung. 2019. Streaming session-based recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1569–1577.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Kalervo Järvelin and Jaana Kekäläinen. 2000. Ir evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 41–48.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *CIKM*, pages 1419–1428.

Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. Stamp: short-term attention/memory priority model for session-based recommendation. In *SIGKDD*, pages 1831–1839.

Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning*, pages 1614–1623. PMLR.

Ben Peters, Vlad Niculae, and André FT Martins. 2019. Sparse sequence-to-sequence models. *arXiv preprint arXiv:1905.05702*.

Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *RecSys*, pages 130–137.

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*, pages 811–820.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295.

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*, pages 555–563.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*, pages 1441–1450.

Constantino Tsallis. 1988. Possible generalization of boltzmann-gibbs statistics. *Journal of statistical physics*, 52(1):479–487.

Ellen M Voorhees. 2001. Overview of the trec 2001 question answering track. In *In Proceedings of the Tenth Text REtrieval Conference (TREC)*. Citeseer.

Weiqing Wang, Hongzhi Yin, Xingzhong Du, Wen Hua, Yongjun Li, and Quoc Viet Hung Nguyen. 2019. Online user representation learning across heterogeneous social networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 545–554.

Xin Wang, Steven CH Hoi, Martin Ester, Jiajun Bu, and Chun Chen. 2017. Learning personalized preference of strong and weak ties for social recommendation. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1601–1610.

Ziyang Wang, Wei Wei, Gao Cong, Xiao-Li Li, Xian-Ling Mao, and Minghui Qiu. 2020. Global context enhanced graph neural networks for session-based recommendation. In *SIGIR*, pages 169–178.

Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 346–353.

Lin Xiao, Zhang Min, Zhang Yongfeng, Liu Yiqun, and Ma Shaoping. 2017. Learning and transferring social and item visibilities for personalized recommendation. In *CIKM*, pages 337–346.

Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A simple convolutional generative network for next item recommendation. In *WSDM*, pages 582–590.

Jiahao Yuan, Zihan Song, Mingyou Sun, Xiaoling Wang, and Wayne Xin Zhao. 2021. Dual sparse attention network for session-based recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4635–4643.

Mengqi Zhang, Shu Wu, Meng Gao, Xin Jiang, Ke Xu, and Liang Wang. 2020. Personalized graph neural networks with attention mechanism for session-aware recommendation. *IEEE Transactions on Knowledge and Data Engineering*.

Tong Zhao, Julian McAuley, and Irwin King. 2014. Leveraging social connections to improve personalized ranking for collaborative filtering. In *CIKM*, pages 261–270.