Dual Sparse Attention Network For Session-based Recommendation

Jiahao Yuan,[1] **Zihan Song,**[1] **Mingyou Sun,**[1] **Xiaoling Wang,**[1, 2*] **Wayne Xin Zhao**[3]

[1] Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai, China
[2] Shanghai Institute of Intelligent Science and Technology, Tongji University, Shanghai, China
[3] Gaoling School of Artificial Intelligence, Renmin University of China
{jhyuan, zhsong, mysun}@stu.ecnu.edu.cn, xlwang@cs.ecnu.edu.cn, batmanfly@gmail.com

Abstract

Session-based Recommendations recommend the next possible item for the user with anonymous sessions, whose challenge is that the user's behavioral preference can only be analyzed in a limited sequence to meet their need. Recent advances evaluate the effectiveness of the attention mechanism in the session-based recommendation. However, two simplifying assumptions are made by most of these attention-based models. One is to regard the last-click as the query vector to denote the user's current preference, and the other is to consider that all items within the session are favorable for the final result, including the effect of unrelated items (i.e., spurious user behaviors). In this paper, we propose a novel **D**ual **S**parse **A**ttention **N**etwork for the session-based recommendation called DSAN to address these shortcomings. In this proposed method, we explore a learned target item embedding to model the user's current preference and apply an adaptively sparse transformation function to eliminate the effect of the unrelated items. Experimental results on two real public datasets show that the proposed method is superior to the state-of-the-art session-based recommendation algorithm in all tests and also demonstrate that not all actions within the session are useful. To make our results reproducible, we have published our code on https://github.com/SamHaoYuan/DSANForAAAI2021.

Introduction

The core of personalized recommendation systems is to recommend different products or services to users by analyzing their past behaviors. Therefore, sufficient historical data and a complete user profile are critical for accurate recommendation results. However, in many practical scenarios, the user often cannot be identified, and their related historical behaviors cannot be associated. In this case, the recommender system needs to accurately capture the user's intent and preference from a relatively short user session and generate the recommended results with the limited information. Thus, the session-based recommender system, which aims to predict the next possible click or consumption item based on the current session, has attracted the attention in academics and industries (Ludewig and Jannach 2018; Zhao et al. 2020).

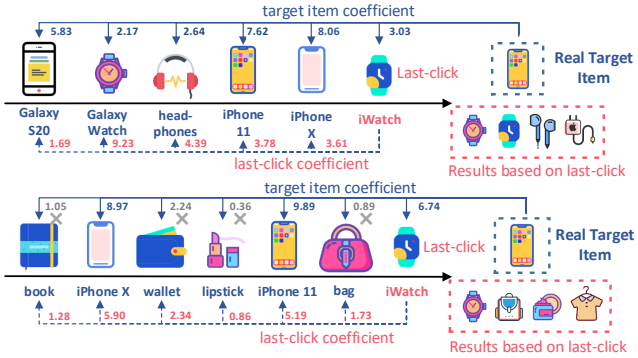


Figure 1: Motivating examples of session-based recommendation. This paper aims to directly model the real target item representation and alleviate the impact of unrelated items.

The critical problem of the session-based recommendation is how to make the best use of limited information to generate accurate recommendation results. Recently, the attention mechanism, which can automatically assign different influence weights to items to capture the related information, has demonstrated outstanding performance in sequence modeling and is widely applied in various session-based recommendation algorithms (Li et al. 2017; Liu et al. 2018; Xu et al. 2019; Luo et al. 2020). The weighted sum of the items within the session gives a better representation of the user preference and improves prediction accuracy, showing the importance of the attention mechanism in this task.

Although the attention-based methods significantly improve the prediction results, there are still some limitations. First, the last-click, which is the last user action within a session, is usually considered as the user's current interest (Liu et al. 2018; Xu et al. 2019; Luo et al. 2020) or used as the query vector for the weight assignment in RNN (Li et al. 2017), MLP (Liu et al. 2018), and GNN models (Wu et al. 2019), but it does not always accurately reflect the user's real preference. For example, as illustrated in Figure 1, if we use the last-click item iWatch as the query vector, the resulting attention coefficient is far from the score of using the item iPhone 11, which is the real target item. Since the real target item cannot be known in advance, an intuitive solution is to use each item in the candidate set as the query

*Corresponding author

vector to find the best one. However, in the real world, the candidate set is vast, and the enormous computational overhead makes this choice unacceptable. Second, not all items in the session are related to the user's intent; for example, clicking one item by mistake or browsing through extraneous promotions. Even if we use the real target item, the conventional attention mechanism still give the unrelated item a small weight, as demonstrated by "book", "wallet", "lipstick", and "bag" in the second session of Figure 1, which may add some useless information to the final representation of the session. Consequently, this strictly positive score is wasteful, making models less interpretable and assigning probability mass to many implausible outputs (Peters, Niculae, and Martins 2019).

To overcome the issue mentioned above, we propose a dual sparse attention network for the session-based recommendation. For simplicity, we name the proposed model DSAN. Specially, we first explore the interaction and correlation between each item within the session and learn a representation of the user's current preference based on the known information by a self-attention network. For convenience, we call this representation **target embedding** since it contains the position information of the target item. Noting that we do not use any other unknown information like the real index of the target item. Subsequently, we use this target embedding as a query vector and apply a vanilla attention network to distinguish the importance of different items within the current session. In this way, we could construct a more reliable session representation. Finally, we combine this learned target embedding and the entire session representation by a neural network to get a final representation, which is used to predict the user's next click. Moreover, to tackle the possible spurious user behaviors in the session, we introduce an adaptively sparse transformation function in both attention layers, which has context-dependent sparsity patterns to pick out more useful items, give higher weight to the critical item, and zero value to the useless item in the session.

The main contributions of this work are summarized as follows:

- We propose a novel framework based on a dual attention network composed of self-attention and vanilla attention. It learns a target embedding with the item-level collaborative information and uses different *query*, *key*, and *value* vectors for high-level information to model the session-based recommendation scenario effectively.
- We introduce an adaptively sparse attention mechanism based on the context to find the possible unrelated item in the session and retain higher weight for the related items.
- Experiments on two real datasets show that our proposed model can not only achieve better results than state-of-the-art methods but also improves the model discernibility because of the sparse attention mechanism.

Related Work

In this section, we review the related work about the session-based recommendation, including conventional methods and neural-network-based methods.

Conventional Methods

Since the user's identification is unknown, the session-based recommendation is limited to the context within the sessions. Hence, for conventional methods, simple matrix factorization (Mnih and Salakhutdinov 2008; Koren and Bell 2015) and Item-KNN (Sarwar et al. 2001) are not suitable for the session-based scene because of ignoring the order of the user's behaviors. To be more consistent with the sequence scenario, Gu, Dong, and Zeng proposed the model based on Markov chains, making predictions by the sequential connections between adjacent clicks. For sequence-aware recommendation tasks, FPMC (Rendle, Freudenthaler, and Schmidt-Thieme 2010) combines Markov chain and matrix factorization to simulate the sequential behavior between two adjacent clicks, which achieves a better result. Recently, methods based on nearest-neighbors obtain competitive performance. SKNN (Jannach and Ludewig 2017) is a session-based k-nearest-neighbors approach, which considers the sessions that contain any item of the current session as neighbors. STAN (Garg et al. 2019) is an extension of SKNN that additionally considers three factors, e.g., the position. However, this kind of approach merely considers the item relevance information and ignores the transition of the user's interest reflected by the session sequences.

Neural-Network-Based Methods

In recent years, thanks to the powerful representation capability of deep learning, neural-network-based methods have made significant progress in the session-based recommendation. Hidasi et al. first introduced the recurrent neural network with gated recurrent unit cell to model user sessions named GRU4Rec. GRU4Rec+ (Tan, Xu, and Liu 2016) proposed a data augmentation method and considered the changes in user behavior over time, further enhancing the prediction results. Moreover, NARM (Li et al. 2017) proposed an encoder-decoder model based on RNN and attention mechanism. STAMP (Liu et al. 2018) introduced a short-term memory priority model based on the multi-layer perceptron and the attention mechanism. These attention-based models separately deal with the user's last-click and the whole session, effectively capturing both the user's general and current interest. CSRM (Wang et al. 2019) and CoSAN (Luo et al. 2020) took collaborative neighborhood information into the session-based recommendation. Then, SR-GNN firstly (Wu et al. 2019) introduced a graph neural network (GNN) into this task. Furthermore, GC-SAN (Xu et al. 2019) proposed a graph contextualized self-attention model, which utilizes both graph neural network and self-attention mechanism. FGNN (Qiu et al. 2019) used a weighted attention graph layer and a Readout function to learn embeddings of items and sessions. With the help of GNN, the model is capable of capturing the complex transition between items and attains better item embeddings, which improves the performance to a great extent.

However, all these neural-network-based methods with the attention model use the last-click or other heuristic ways to generate the query vector, which ignores the real user's current preference. Bert4Rec (Sun et al. 2019) employs the

deep bidirectional self-attention to model user behaviors and first tries to learn the user's current preferred target embedding, but the simple repetitive multi-layer structure makes the model lose the original information.

Preliminary

In this section, we formulate the problem of session-based recommendation and then introduce the sparse transformation function, which serves as a fundamental module of our proposed model.

Problem Statement

The typical task of session-based recommendation is to predict the item that the user most likely interact with next based solely on the action records (e.g., clicks) within a short period. Since the user's long-term preference profile is unknown, making decisions with limited information is a considerable challenge.

Let $I = \{i_1, i_2, ..., i_m\}$ denote the set of all unique items involved in all sessions and $S = \{s_1, s_2, ..., s_n\}$ denote a session ordered by timestamps, where $s_p \in I$ represents the p^{th} clicked item of the user and n is the length of the session. In this paper, given a prefix of the session truncated at time step t, $S_t = \{s_1, s_2, ..., s_t\}(1 < t < n)$, the goal of the proposed model is to predict the next click item s_{t+1}. To be exact, the proposed model learns to generate a score $\hat{y}_i$ for each item $i \in I$ and then items corresponding to the top-K scores will be presented to users as results.

Sparse Transformation Function

The two core components of the attention mechanism are the alignment model and the transformation function. One is used to compute attention weights, and the other is to transform weights into probabilities. Usually, the transformation function is a well-known function **softmax** (Bridle 1990), which returns positive values and dense output probabilities. However, this nonzero probability may assign weights to the useless data, affecting the ability to find the relevant items.

Essentially, softmax is one of mappings from $\mathbb{R}^d$ to $\triangle^{d-1}$, where $\triangle^{d-1} = \{p \in \mathbb{R}^d | 1^T p = 1, p \geq 0\}$ denotes the d-1 dimensional simplex. Sparse transformations tend to yield zero for the low-scoring in the vector, which is named **sparsemax** (Martins and Astudillo 2016):

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

where x is the input vector and p is the output vector. In this paper, we introduce a novel family of the transformation function, namely **α-entmax** (Peters, Niculae, and Martins 2019), to replace the softmax, which has been proven useful for the application of NLP (Garg et al. 2019; Correia, Niculae, and Martins 2019),

$$\alpha\text{-entmax}(x) = \underset{p \in \triangle^{d-1}}{\operatorname{argmax}} p^T x + H_\alpha^T(p), where$$

$$H_\alpha^T(p) = \begin{cases} \dfrac{1}{\alpha(\alpha-1)} \sum_j (p_j - p_j^\alpha), & \alpha \neq 1 \\ H^S(p), & \alpha = 1 \end{cases} \tag{2}$$

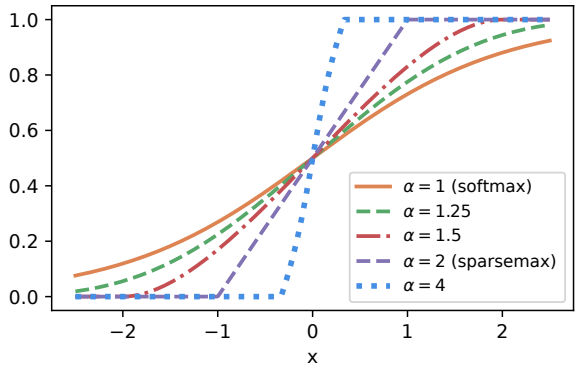


Figure 2: Illustration of α-entmax in the two-dimensional case.

where $H_\alpha^T(p)$ is the Tsallis α-entropies (Tsallis 1988), which is a family of entropies parametrized by a scalar $\alpha > 1$. From the equation 2, we can prove that the softmax function equals 1-entmax and sparsemax is the 2-entmax, where the Shannon entropy and Gini entropy are the entropic regularizers respectively. As illustrated in Figure 2, the parameter α controls the shape and sparsity of this function. When $1 < \alpha < 2$, this function tends to produce sparse probability distribution and has smooth corners. In this paper, in order to distinguish unrelated information in the session, we replace the transformation function with α-entmax in the attention mechanism. Moreover, we present a method to automatically learn α, allowing each session to choose the parameter based on the context adaptively.

Method

In this section, we present the proposed model in detail. This model has four main components: embedding layer to convert input session into two vectors, target embedding learning to exploit the item-level collaborative information by sparse self-attention, target attention layer to combine with initial information, and prediction layer for the final result. The complete pipeline is demonstrated in Figure 3.

Embedding Layer

First of all, we present an embedding layer to convert the input session into two vectors. One is the item embedding, and the other is the positional embedding. We introduce a learnable positional embedding module (Sun et al. 2019), which is used to map the position index to a dense vector for capturing the temporal influence of the input. Formally, given the input session $S_t = \{s_1, s_2, ..., s_t\}$, for any element $s_p \in S_t$, the hidden representation of it is

$$c_i = Concat(x_i, p_i) \tag{3}$$

where $x_i \in \mathbb{R}^d$ is the embedding of the item, $p_i \in \mathbb{R}^d$ is the positional embedding of the item, and $c_i \in \mathbb{R}^{2d}$ is the concatenated embedding of the item and position. Positional embeddings allow the proposed model to know about the portion of the session that it is dealing with, so the concatenated embedding could learn about the item's collaborative information of the item and position within the session.

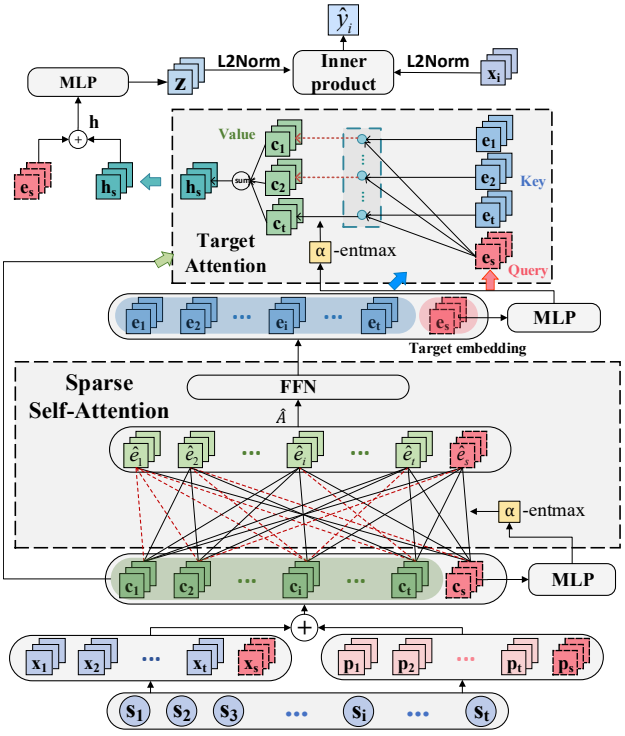


Figure 3: The general architecture of the proposed model. The red dot line indicates a possible zero weight value.

To learn the current user's preference representation without specifying the target item information, we append a special item index (e.g., the largest item index plus one) at the end of the input sequence, which is denoted by x_s with its position embedding p_s and is initialized with other item embeddings, to indicate the item that we need to predict. So the concatenated embedding is $\hat{C} = \{c_1, c_2, ..., c_t, c_s\}$. c_s, at the position of $t+1$, is composed of x_s and p_s, including the information of the special item index and the position of the item to be predicted. The special item index in different sessions share the same embedding.

Target Embedding Learning

As illustrated in Figure 3, the outputs of the embedding layer are the concatenated embeddings $\hat{C}$. Then we introduce how to learn the target embedding by self-attention to fuse the entire context of the session. Firstly, we adopt the *Scaled Dot-Product Attention* with the sparse transformation to capture the dependency between item pairs within the session:

$$\hat{A} = \alpha\text{-entmax}(\frac{QK^T}{\sqrt{2d}})V \tag{4}$$

where Q contains representations of the queries, K is the key matrix, and V is the value matrix of the items attended. In this layer, the sparse transformation is used to generate the attention weight for the first time to ensure that the learned target embedding and new item embeddings contain less unrelated information as much as possible. For each session, we learn its own α by:

$$\alpha = \sigma(W_\alpha c_s + b_\alpha) + 1 \tag{5}$$

where $W_\alpha \in \mathbb{R}^{1\times 2d}$ denotes the weighting matrix, $b \in \mathbb{R}$ is the bias value, and σ denote the sigmoid function. In this way, α is determined by c_s in different session, which included the special item index and the largest position. The value of α is also mapped to $[1, 2]$, which has a smooth corner. Then we make $K = V = \hat{C}$, but

$$Q = f(\hat{C}W^Q + b^Q) \tag{6}$$

where $W^Q \in \mathbb{R}^{2d\times 2d}$ is the weighting matrix, $b^Q \in \mathbb{R}^{2d}$ is the bias vector and $f(\cdot)$ denotes activate function ReLU. Through the projection, the representation of the item can be more flexible. For example, the dot product in self-attention satisfies the commutative law, that is, QK^T is equal to KQ^T if $Q = K$, leading to different roles to be undifferentiated.

Although the self-attention mechanism learns the new representation of all items, it is mainly based on linear projections. Then we apply *Position-wise Feed-Forward Network* to endow the model with more non-linearity,

$$FFN(\hat{A}) = \max(0, \hat{A}W_1^{self} + b_1)W_2^{self} + b_2 \tag{7}$$

where $W_1^{self}, W_2^{self} \in \mathbb{R}^{2d\times 2d}$ are both weighting matrices, $b_1, b_2 \in \mathbb{R}^{2d}$ are bias vectors, and all sessions will share the same parameters. After that, we add a residual connection and layer normalization on the result to alleviate the instability of the model training. We also add the dropout mechanism to alleviate the overfitting. For simplicity, we define the whole sparse self-attention network above as

$$E = SAN(\hat{C}) \tag{8}$$

where $E = \{e_1, e_2, ..., e_t, e_s\}$ is the final output of self-attention network. Additionally, $\{e_1, e_2, ..., e_t\}$ are new item embeddings of the session after the information extraction by the network. e_s is the learned target embedding, which contains the special item index and fuses the entire session's information to denote the user's real current preference.

Target Attention Layer

The self-attention network can be considered as a feature extractor, in which each item embedding contains the information with each other, including the learned target embedding. However, it ignores the initial information after the information extraction.

Therefore, in this layer, we directly apply a vanilla attention network based on the different query, key, and value to learn the entire session representation. Given $E \in \mathbb{R}^{|\hat{C}|\times 2d}$, the target attention layer aims to learn the whole session representation based on the learned target embedding and the initial input. Formally, a feed-forward network is used to learn the weight:

$$\beta_p = \alpha\text{-entmax}(W_0 f(W_1 e_p + W_2 e_s + b_a)) \tag{9}$$

where $W_1, W_2 \in \mathbb{R}^{2d\times 2d}, W_0 \in \mathbb{R}^{1\times 2d}$ are the weighting matrices, $b_a \in \mathbb{R}^{2d}$ is the bias vector, $f(\cdot)$ denotes the activation function ReLU, and α is given by

$$\alpha = \sigma(W_\alpha e_s + b_\alpha) + 1. \tag{10}$$

Noting that $K = \{e_1, e_2, ..., e_t\}$ is the key matrix and e_s is the vector of query, which are both the outputs of the equation 8. Here we use the sparse transformation again to get rid of the unrelated items in the initial input session when learning the whole session representation.

In a word, we use the learned target embedding to be the query, and the new item embeddings learned by the self-attention network to be the key. β_p represents the attention weight of the item e_p with the learned query item e_s. As a result, this attention weight is based on the known information of all clicked items and the target item. It can capture the correlations between the item within the session and the user's current preference.

After obtaining the attention score vector $\beta = \{\beta_1, \beta_2, ..., \beta_t\}$ with respect to the current session prefix S_t, the session representation can be calculated as $h_s = \sum_{p=1} \beta_p c_p$, where $h_s \in \mathbb{R}^{2d}$ denotes the session representation and c_p is the value vector, which is initial item embedding before entering into the self-attention network.

Prediction Layer

In this layer, we evaluate the probability of the next clicking item based on the outputs above. We first concatenate the learned target embedding e_s and the session embedding h_s, and then apply a feed-forward neural network to get the final representation of the proposed model,

$$\begin{aligned} h &= Concat(e_s, h_s) \\ z &= f(W_z h + b_z) \end{aligned} \tag{11}$$

where $z \in \mathbb{R}^d$ denotes the final output of the proposed model, $W_z \in \mathbb{R}^{d \times 4d}$ is the weighting matrices, $b_z \in \mathbb{R}^d$ is the bias vector and $f(\cdot)$ is the non-linear activation function SELU (Klambauer et al. 2017). For each item $i \in I$, we get its probability as follows:

$$\begin{aligned} \hat{z} = w_k L2Norm(z),\ \hat{x_i} = L2Norm(x_i) \\ \hat{y_i} = softmax(\hat{z}^T \hat{x_i}) \end{aligned} \tag{12}$$

where x_i is the initial embedding of the item i and $\hat{y_i}$ denotes the probability of the item in the candidate item set I. $L2Norm$ is the L2 Normalization function and w_k is the normalized weight. This weighted normalization (Gupta et al. 2019) and the Regularizing Softmax loss (Zheng, Pal, and Savvides 2018) make the training process more stable and insensitive to hyper-parameters.

Finally, the loss function is defined as the cross-entropy of the prediction and the ground-truth. It can be written as follows:

$$L(y, \hat{y}) = -\sum_{i=1}^{m} y_i \log(\hat{y_i}) \tag{13}$$

where y is the one-hot encoding vector of the ground truth.

Experiments and Analysis

In this section, we first describe the setup of the experiments. And then, we design experiments to prove the performance of our proposed DSAN and conduct detailed analysis under different experimental settings.

Datesets	# train	# test	# clicks	# items
Diginetica	526,135	44,279	858,108	40,840
Retailrocket	433,648	15,132	710,586	36,968

Table 1: Statistics of datasets used in the experiments

Experiments Setup

Datasets To evaluate the effectiveness of the proposed model, we use two real-world representative datasets, i.e., Diginetica[1] and Retailrocket[2]. For simplicity, we name them DN and RR. The DN dataset comes from CIKM Cup 2016, and we only used the released transaction data. The RR is a dataset on a Kaggle contest published by an e-commerce company, which contains the user's browsing activity within six months. Both two datasets are publicly available.

For a fair comparison, we follow the previous work (Xu et al. 2019) to filter out all sessions of length less than 3 and items which are fewer than 5 occurrences in each dataset, and then we take last week in the two datasets as the test set. Furthermore, we conducted a segmentation preprocessing for the session. For each input session $S = \{s_1, s_2, ..., s_n\}$, we have generated the corresponding session prefix and label pairs $([s_1, s_2]; s_3), ([s_1, s_2, s_3]; s_4), ..., ([s_1, s_2, ..., s_{n-1}]; s_n)$. It is worth noting that the session containing at least two items is more conducive to methods based on the GNN (Xu et al. 2019). The statistics of the three datasets after preprocessing are shown in Table 1.

Baselines and Metrics We compare the proposed model DSAN with the following methods:

- **S-POP** recommends the most popular items in the current session. It is an improved version of the popularity-based method.
- **FPMC** (Rendle, Freudenthaler, and Schmidt-Thieme 2010) is a hybrid model combining matrix factorization and Markov chains.
- **SKNN** (Jannach and Ludewig 2017) is a session-based k-nearest-neighbors approach.
- **STAN** (Garg et al. 2019) is an extension of SKNN approach with some additional factors.
- **GRU4Rec** (Hidasi et al. 2016) is a session-based recommendation model based on GRU layers.
- **STAMP** (Liu et al. 2018) is a short-term memory priority model, which exploits the user's current interest reflected by the last-click.
- **SR-GNN** (Wu et al. 2019) is a session-based recommendation model that applies a graph neural network to learn the item and session representation.
- **GC-SAN** (Xu et al. 2019) makes recommendations by combining a single-layer graph neural network and multi-layer self-attention network.

[1] http://cikm2016.cs.iupui.edu/cikm-cup

[2] https://www.kaggle.com/retailrocket/ecommerce-dataset

Datasets	Diginetica				Retailrocket			
Metrics	HR@10	HR@20	MRR@10	MRR@20	HR@10	HR@20	MRR@10	MRR@20
S-POP	0.2389	0.2409	0.1392	0.1393	0.3578	0.3803	0.2468	0.2481
FPMC	0.1807	0.2571	0.0713	0.0765	0.2599	0.3237	0.1338	0.1382
SKNN	0.3661	0.4835	0.1561	0.1649	<u>0.4674</u>	<u>0.5428</u>	0.2593	0.2646
STAN	<u>0.3820</u>	0.4993	0.1678	0.1759	0.4656	0.5348	0.2633	0.2681
GRU4Rec	0.2617	0.3927	0.0969	0.1059	0.3835	0.4401	0.2327	0.2367
STAMP	0.3349	0.4647	0.1399	0.1489	0.4295	0.5096	0.2461	0.2517
SR-GNN	0.3772	0.5050	0.1675	0.1763	0.4321	0.5032	0.2607	0.2657
GC-SAN	0.3786	<u>0.5084</u>	<u>0.1689</u>	<u>0.1779</u>	0.4410	0.5118	<u>0.2692</u>	<u>0.2740</u>
Bert4Rec	0.3461	0.4878	0.1327	0.1425	0.4585	0.5419	0.2584	0.2642
CoSAN	0.3475	0.4834	0.1429	0.1522	0.4381	0.5247	0.2380	0.2440
DSAN	**0.4029***	**0.5376***	**0.1805***	**0.1899***	**0.4905***	**0.5654***	**0.3021***	**0.3074***
Improv.	5.47%	5.74%	6.87%	6.75%	4.75%	3.78%	10.70%	11.28%

Table 2: Performance of all recommendation models. The boldface is the best result over all methods, the underline is the best result of all baselines, and * denotes the significant difference for t-test.

- **Bert4Rec** (Sun et al. 2019) is a method that employ the deep bidirectional self-attention to model user behaviors for sequential recommendation.
- **CoSAN** (Luo et al. 2020) learn the session representation and predict the intent of the current session by investigating neighborhood sessions.

We adopt two common metrics on session-based recommendation in this paper, Hit Rate(HR@K) and Mean Reciprocal Rank (MRR@K). HR is the proportion of cases when the ground truth is ranked amongst the top-K items. It is used to evaluate unranked results. MRR is the average of reciprocal ranks of the desired items, which is the evaluation of ranked results. In this study, we consider the Top-K ($K = 10, 20$) results.

Parameters setup In this paper, all hyper-parameters are optimized via grid search on each dataset, respectively. According to the experimental results, the optimal hyper-parameters are $\{\eta : 0.001, \varepsilon : 0.5, w_k : 20\}$ on two datasets, where η is the learning rate, ε is the dropout rate and w_k is the normalized weight. We use Adam as the model optimizer, and explore the case that embedding dimension $d = 100$ for a fair comparison, which is the hyper-parameter in the previous work. We implement the model by Pytorch, and the mini-batch settings are $\{$batch size: 512, epoch: 50$\}$.

Results and Discussion

Overall performance To demonstrate the recommendation performance of our proposed model **DSAN**, we compare it with other state-of-the-art baselines. The experimental results on two datasets are shown in Table 2.

The first four methods, S-POP, FPMC, SKNN and STAN, are all improved conventional methods, but they have achieved significant results. Firstly, the performance of FPMC is poor since we consider each session as a user but do not have the long-term user profile. However, SKNN and STAN reach competitive results compared with neural-networks methods, especially on RR dataset. They are all improved version of the KNN method, proving that the conventional methods are not necessarily weaker than the neural-network-based methods when they incorporate with the session-level information.

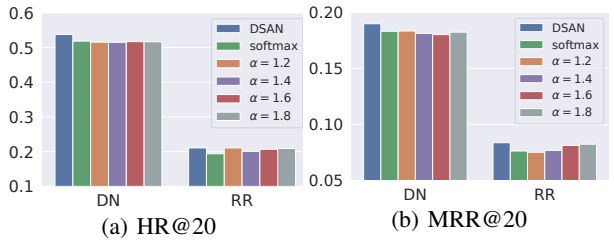
Neural-network-based models can generally achieve better results except for GRU4Rec. GRU4Rec, the first proposed neural-network-based method to solve session-based recommendations, performs worst among all neural-network methods. Furthermore, the STAMP model, which considers long-term and short-term memory fusion in the session and especially regards the last-click as the user's current preference, reaches a not very prominent result on RR dataset. SR-GNN and GC-SAN both construct graphs for sessions, but GC-SAN incorporates self-attention networks. Therefore, the performance of GC-SAN is better than SR-GNN, since it combines the long-range self-attention representation and the short-term interest of the last-click. Bert4Rec also has a learned embedding, including the known positional information, but they directly use the last layer to present the whole session, ignoring the initial information. As we can see, its result is competitive but still inferior to our proposed model. Similarly, CoSAN combines the self-attention network and the session-level collaborative information, which performs equally well or significantly better in some cases than other neural-network-based methods. It is noted that the three models with the self-attention network all have competitive performance, demonstrating the importance of this feature extraction method.

Our proposed model DSAN outperforms all baselines significantly. Compared to Bert4Rec, we introduce a dual attention network, which is applicable to different situations. Compared with CoSAN and GC-SAN, DSAN adopts the self-attention mechanism for the whole session representation and a learned target embedding to be the query vector for the user's real intent, leveraging all known information for the final representation. Without investigating neighborhood sessions and adopting the graph neural network, we can still achieve the best results.

Effect of the dual attention network In order to verify the effect of each module, we design three contrast models for comparison: **DSAN-NS** does not have the target embed-

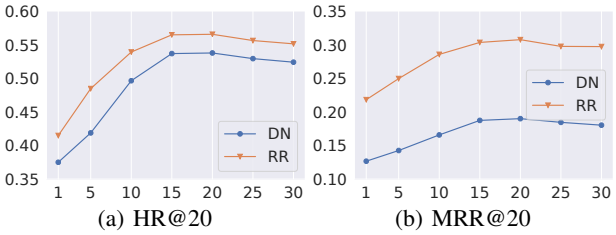
Datesets	DIGINETICA		RETAILROCKET	
Metrics	HR@20	MRR@20	HR@20	MRR@20
DSAN-NS	0.5199	0.1858	0.5322	0.3000
DSAN-NT	0.5348	0.1838	0.5646	0.3040
DSAN-DA	0.5340	0.1876	0.5600	0.3010
DSAN	**0.5376**	**0.1899**	**0.5654**	**0.3074**

Table 3: Impacts of the dual attention network.



(a) HR@20 (b) MRR@20

Figure 4: Experimental results with different transformation function on two metrics.



(a) HR@20 (b) MRR@20

Figure 5: Performance with different normalized weight w_k.

ding learning and uses the last-click as the query in a vanilla attention network. **DSAN-NT** does not have the target attention layer, which only uses the learned target embedding from the self-attention network as the whole session representation. **DSAN-DA** separates self-attention and vanilla attention into two parallel structures, which also use the last-click as a query in the vanilla attention layer.

As shown in Table 3, we report the result on HR@20 and MRR@20. First of all, we can observe that DSAN-NS has the worst performance in most cases, indicating that the last-click does not exhibit the user's real intent and interest in all cases. In contrast, DSAN-NT has the second better performance, proving that the learned target embedding positively contributes to the result of a given session. DSAN-DA is better than DSAN-NS but worse than DSAN-NT in general, which further proves the problem of using the last-click. Both DSAN-NS and DSAN-NT only have one single attention network, and their performances are worse than the proposed model DSAN, indicating the importance of the dual attention module; indeed, it can get a better representation of the whole session. Also, the fact that DSAN performs best in all cases demonstrates the advantages of considering both the dual attention network and the learned target embedding. This experiment is also an ablation experiment for our proposed model, which proves that each module in the model structure is an indispensable content.

Comparison with different transformation function To demonstrate the utility of α-entmax, we compare the experimental results on two datasets using the proposed adaptive α and the fixed α. The range of fixed α is in $\{1, 1.2, 1.4, 1.6, 1.8\}$. We set the minimum session length of 15 for RR since the longer session is more likely to contain unrelated click information. As illustrated in Figure 4, the performance of the softmax function, which is $\alpha = 1$, is relatively poor, especially in RR. We infer that the probability of users being affected by the external environment significantly increases in the longer session, so that the longer session may have more unrelated items, but the softmax function still gives small weight to them. The performance based on the adaptively sparse transformation is better than fixed α, indicating that each session has its own best α based on the context. Moreover, the fact that the performance of fixed α is close to softmax in DN has further proved this. Therefore, the sparse transformation, which can effectively improve the prediction accuracy, is of great significance.

Influence of the normalized weight w_k In order to explore the stability of DSAN, we explore the influence of the hyper-parameter w_k. Since the popularity bias and cross-entropy based on dot product, target items with higher L^2 norm easier to be predicted. However, when we use L2 Normalization function, the cosine similarity is restricted to $[-1, 1]$. A scaling factor w_k is useful in practice to allow for better convergence, which gets saturated at high values for the training set. Compared to all baselines, this is a unique hyper-parameter of the proposed model.

As shown in Figure 5, the effectiveness of the proposed model is extremely related to this parameter. When $w_k = 1$, it is equivalent to the cosine similarity and the model achieves the worst result, indicating that it is not a good measure. As w_k increases, the result gradually improves on both datasets, and they both achieve the best result when $w_k = 20$, proving that the appropriate value can make the training of this model more stable to get better performance. With w_k further increase, the result has declined since a too larger value will lead to overfitting and thus affect the proposed model's performance.

Conclusion and Future Work

In this paper, we propose a dual sparse attention network for session-based recommendation. Specifically, we first use a self-attention network with the positional embedding to generate the target item embedding and then integrate a vanilla attention network to learn the entire session representation. Next, we combine the two vectors to rank all items. We also introduce a new adaptively sparse transformation function to replace the softmax, making useful information more focused. Extensive experimental analysis verified that our proposed model DSAN is superior to state-of-the-art methods. In the Future work, we plan to explore the way of using automated machine learning to delete the unrelated item in the session before entering into the network, so that the model can learn more useful information.

Acknowledgements

This work was supported by NSFC grants (No. 61532021 and 61972155), the Science and Technology Commission of Shanghai Municipality (20DZ1100300 and 19511120200) and the Open Project Fund from Shenzhen Institute of Artificial Intelligence and Robotics for Society.

Ethics Statement

We promise this paper is our original work and no portion of this paper has been previously published. Our work is mainly focus on theoretical area of the recommender system based on anonymous users, so that we think it will not make any directly negative societal implications. The positive implication is that we aim to help ease the information overload of the recommender system and reduce the time for users to find desired items.

References

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

Correia, G. M.; Niculae, V.; and Martins, A. F. T. 2019. Adaptively Sparse Transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2174–2184. Hong Kong, China: Association for Computational Linguistics.

Garg, D.; Gupta, P.; Malhotra, P.; Vig, L.; and Shroff, G. 2019. Sequence and time aware neighborhood for session-based recommendations: Stan. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1069–1072.

Gu, W.; Dong, S.; and Zeng, Z. 2014. Increasing recommended effectiveness with markov chains and purchase intervals. *Neural Computing and Applications* 25(5): 1153–1162.

Gupta, P.; Garg, D.; Malhotra, P.; Vig, L.; and Shroff, G. 2019. NISER: Normalized Item and Session Representations to Handle Popularity Bias. *arXiv* arXiv–1909.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2016. Session-based recommendations with recurrent neural networks. In *ICLR*.

Jannach, D.; and Ludewig, M. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, 306–310. ACM.

Klambauer, G.; Unterthiner, T.; Mayr, A.; and Hochreiter, S. 2017. Self-normalizing neural networks. In *Advances in neural information processing systems*, 971–980.

Koren, Y.; and Bell, R. 2015. Advances in collaborative filtering. In *Recommender systems handbook*, 77–118. Springer.

Li, J.; Ren, P.; Chen, Z.; Ren, Z.; Lian, T.; and Ma, J. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 1419–1428. ACM.

Liu, Q.; Zeng, Y.; Mokhosi, R.; and Zhang, H. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1831–1839. ACM.

Ludewig, M.; and Jannach, D. 2018. Evaluation of session-based recommendation algorithms. *User Modeling and User-Adapted Interaction* 28(4-5): 331–390.

Luo, A.; Zhao, P.; Liu, Y.; Zhuang, F.; Wang, D.; Xu, J.; Fang, J.; and Sheng, V. S. 2020. Collaborative Self-Attention Network for Session-based Recommendation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 2591–2597.

Martins, A.; and Astudillo, R. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International Conference on Machine Learning*, 1614–1623.

Mnih, A.; and Salakhutdinov, R. R. 2008. Probabilistic matrix factorization. In *Advances in neural information processing systems*, 1257–1264.

Peters, B.; Niculae, V.; and Martins, A. F. T. 2019. Sparse Sequence-to-Sequence Models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 1504–1519. Florence, Italy: Association for Computational Linguistics.

Qiu, R.; Li, J.; Huang, Z.; and Yin, H. 2019. Rethinking the Item Order in Session-based Recommendation with Graph Neural Networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 579–588.

Rendle, S.; Freudenthaler, C.; and Schmidt-Thieme, L. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th international conference on World wide web*, 811–820. ACM.

Sarwar, B.; Karypis, G.; Konstan, J.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, 285–295.

Sun, F.; Liu, J.; Wu, J.; Pei, C.; Lin, X.; Ou, W.; and Jiang, P. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, 1441–1450.

Tan, Y. K.; Xu, X.; and Liu, Y. 2016. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 17–22. ACM.

Tsallis, C. 1988. Possible generalization of Boltzmann-Gibbs statistics. *Journal of statistical physics* 52(1-2): 479–487.

Wang, M.; Ren, P.; Mei, L.; Chen, Z.; Ma, J.; and de Rijke, M. 2019. A Collaborative Session-Based Recommendation Approach with Parallel Memory Modules. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, 345–354.

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 AAAI Conference on Artificial Intelligence*, volume 33, 346–353.

Xu, C.; Zhao, P.; Liu, Y.; Sheng, V. S.; Xu, J.; Zhuang, F.; Fang, J.; and Zhou, X. 2019. Graph contextualized self-attention network for session-based recommendation. In *Proc. 28th Int. Joint Conf. Artif. Intell.(IJCAI)*, 3940–3946.

Zhao, W. X.; Mu, S.; Hou, Y.; Lin, Z.; Li, K.; Chen, Y.; Lu, Y.; Wang, H.; Tian, C.; Pan, X.; Min, Y.; Feng, Z.; Fan, X.; Chen, X.; Wang, P.; Ji, W.; Li, Y.; Wang, X.; and Wen, J. 2020. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. *ArXiv* abs/2011.01731.

Zheng, Y.; Pal, D. K.; and Savvides, M. 2018. Ring loss: Convex feature normalization for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5089–5097.