Lighter and Better: Low-Rank Decomposed Self-Attention Networks for Next-Item Recommendation

Xinyan Fan[1,2], Zheng Liu[3*], Jianxun Lian[3], Wayne Xin Zhao[1,2*], Xing Xie[3], and Ji-Rong Wen[1,2]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Beijing Key Laboratory of Big Data Management and Analysis Methods
[3]Microsoft Research Asia
{xinyan.fan, jrwen}@ruc.edu.cn, batmanfly@gmail.com, {zhengliu, jianxun.lian, xingx}@microsoft.com

ABSTRACT

Self-attention networks (SANs) have been intensively applied for sequential recommenders, but they are limited due to: (1) the quadratic complexity and vulnerability to over-parameterization in self-attention; (2) inaccurate modeling of sequential relations between items due to the implicit position encoding. In this work, we propose the low-rank decomposed self-attention networks (**LightSANs**) to overcome these problems. Particularly, we introduce the low-rank decomposed self-attention, which projects user's historical items into a small constant number of latent interests and leverages item-to-interest interaction to generate the context-aware representation. It scales linearly w.r.t. the user's historical sequence length in terms of time and space, and is more resilient to over-parameterization. Besides, we design the decoupled position encoding, which models the sequential relations between items more precisely. Extensive experimental studies are carried out on three real-world datasets, where LightSANs outperform the existing SANs-based recommenders in terms of both effectiveness and efficiency.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Low-Rank Self-Attention, Next-Item Recommendation

ACM Reference Format:
Xinyan Fan[1,2], Zheng Liu[3*], Jianxun Lian[3], Wayne Xin Zhao[1,2*], Xing Xie[3], and Ji-Rong Wen[1,2]. 2021. Lighter and Better: Low-Rank Decomposed Self-Attention Networks for Next-Item Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3404835.3462978

1 INTRODUCTION

Sequential recommendations have received increasing interest recently, due to their broad applicability in many online services (e.g., e-commerce and movies). Various methods based on RNN and CNN have been designed for sequential recommenders (e.g., GRU4Rec [4] and Caser [13]). In recent years, self-attention networks (SANs) turn out to be more promising options, as they can model sequential dynamics from user's historical behaviors better by letting the items fully attend to the context (all items that the user interacted with in the past). However, the SANs-based recommenders (e.g., SASRec [5] and BERT4Rec [12]) have two major shortcomings.

• SANs require user's historical items to directly attend to each other (called *item-to-item interaction*), which needs time and space that grows quadratically with historical sequence length [1]. Thus, the running cost may be prohibitive in practice. Besides, direct item-to-item interaction is also vulnerable to *over-parameterization*[1]. A typical sequential recommender involves the modeling of massive items. However, many items do not have sufficient interactions due to the items' long-tail property. Therefore, infrequent items' embeddings will not be well trained, whose related attention weights (generated by item-to-item interaction) can be inaccurate.

• In vanilla SANs, the item embeddings and position embeddings are directly added up. The recent work [7] shows that there are no strong correlations between the item and the absolute position. Thus, such a treatment may introduce noisy correlations and limit the model's capability of capturing the user's sequential patterns.

Some prior works such as Linformer [16] and Performer [2] have attempted to improve the efficiency of SANs. We argue that the existing methods mainly focus on acceleration in general. However, without considering the characteristics of user behaviors, their performances might be limited in recommendation scenarios.

In this work, we propose a novel approach **LightSANs**, which leverage the low-rank property of user history for acceleration. Particularly, we assume that the majority of user's historical items can be categorized with no more than k (a small constant) *latent interests* (the latent interest represents user preference towards a certain group of items; in this work, it is a vector generated from the sequence of user's item embeddings). Based on this property, we introduce the low-rank decomposed self-attention: the user history is projected into k latent interests, and each of the user's historical items merely needs to interact with the k latent interests to establish its context-awareness (called *item-to-interest interaction*). It makes SANs' time and space complexities linear w.r.t. the length of user history. Meanwhile, it avoids item-to-item interaction, which makes the model more resilient to over-parameterization. On the

*Corresponding authors.

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3462978

[1]Over-parameterization usually means the situation in which the amount of information is insufficient to estimate a large number of parameters of deep networks, which leads to inaccuracy and high cost for model's inference [3, 9].

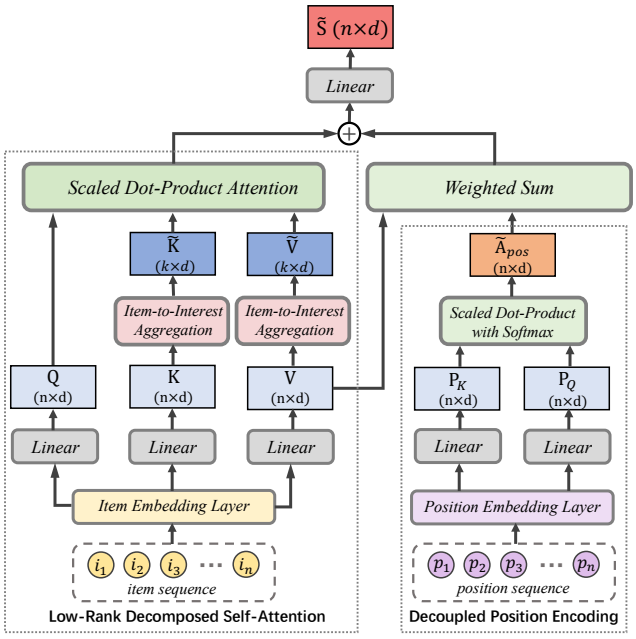


Figure 1: The framework of LightSANs.

other hand, we calculate position correlations individually by our proposed decoupled position encoding, which explicitly models sequential relations between items. Therefore, it benefits the modeling of user's sequential patterns by eliminating noisy correlations.

Our main contributions are summarized as follows:

• A novel SANs-based sequential recommender, LightSANs, with two advantages: (1) the low-rank decomposed self-attention for more efficient and precise modeling of context-aware representations; (2) the decoupled position encoding for more effective modeling of sequential relations between items.

• Extensive experiments on three benchmark recommendation datasets, where LightSANs outperform various SANs-based methods in terms of both effectiveness and efficiency.

2 APPROACH

We focus on the next-item recommendation in this work. Given the ordered sequence of user u's historical items up to the timestamp-t: $\{i_1^u, \cdots, i_t^u\}$, we need to predict the next item, i.e., i_{t+1}^u. We aim to enhance the classical SANs to make them lighter and better. Specifically, we propose LightSANs, which leverage low-rank decomposed self-attention for precise modeling of items' relevance, and decoupled position encoding for explicit modeling of items' sequential relations. The overall framework of LightSANs is depicted in Figure 1, and the details will be introduced next.

2.1 Low-Rank Decomposed Self-Attention

We use the low-rank decomposed self-attention to generate context-aware representations. It projects items into k latent interests and integrates the item with the context through interacting with latent interests. Such a workflow reduces the complexity from $O(n^2)$ to $O(nk)$ and effectively mitigates the over-parameterization problem.

2.1.1 Item-to-Interest Aggregation. We assume that the majority of user's historical items can be categorized with no more than k (a small constant) latent interests. Thus, we propose a learnable projection function $f : \mathbb{R}^{n\times d} \to \mathbb{R}^{k\times d}$, to aggregate the historical items into latent interests. Given the item embedding matrix $\mathbf{H} \in \mathbb{R}^{n\times d}$ (n: #items, d: cardinality of hidden-dimension) as input, we first compute the item-to-interest relevance distribution $\mathbf{D} \in \mathbb{R}^{n\times k}$:

$$\mathbf{D} = \text{softmax}(\mathbf{H} \cdot \mathbf{\Theta}^\top), \tag{1}$$

where $\mathbf{\Theta} \in \mathbb{R}^{k\times d}$ is a learnable parameter. Then, we use the distribution $\mathbf{D}$ to aggregate the input item embedding matrix, and obtain the interest representation matrix $\widetilde{\mathbf{H}} \in \mathbb{R}^{k\times d}$:

$$\widetilde{\mathbf{H}} = f(\mathbf{H}) = \mathbf{D}^\top \cdot \mathbf{H} = \left(\text{softmax}(\mathbf{H} \cdot \mathbf{\Theta}^\top)\right)^\top \cdot \mathbf{H}. \tag{2}$$

Firstly, thanks to f, the item embedding matrix $\mathbf{H}$ ($n \times d$) is converted to the low-rank interest representation $\widetilde{\mathbf{H}}$ ($k \times d$). This aggregation will help decrease the size of the attention matrix effectively, and thus the feed-forward pass of the networks will become more efficient. Besides, interaction with the latent interests can be more reliable than the direct attention to other items, because the latent interests capture the user's overall preferences reflected by the item sequence, according to Eq. 2. As a result, attention weights related to infrequent items under item-to-interest interaction will be more accurate, which mitigates the over-parameterization issue.

Our item-to-interest aggregation is similar to the low-rank linear mapping of Linformer [16] in format. However, there are two main differences between them. Firstly, our model is parameterized with a $(k \times d)$-dimensional matrix $\mathbf{\Theta}$, while the mapping matrix in Linformer is $(n \times k)$-dimension. We argue that a fixed value of n in parameters makes it hard for the model to be adapted to varying sequence length. Secondly, item-to-interest aggregation projects items to the latent interest space through a learnable item-to-interest relevance distribution while Linformer achieves it through a direct linear mapping. We empirically demonstrate that the performance based on linear mapping is limited because items and latent interests do not follow a simple linear relation (Section 3.2.1).

2.1.2 Item-to-Interest Interaction. We convert the input embedding sequence $\mathbf{X}$ into three matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{n\times d}$ through linear projections $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d}$, and feed them into our low-rank decomposed self-attention. The original $\mathbf{K}$ and $\mathbf{V}$ ($n \times d$) in the vanilla multi-head self-attention are mapped into $\widetilde{\mathbf{K}}$ and $\widetilde{\mathbf{V}}$ ($k \times d$) via item-to-interest aggregation f:

$$\widetilde{\mathbf{S}}_i = \text{softmax}\left(\frac{\mathbf{Q}_i \cdot \widetilde{\mathbf{K}}_i^\top}{\sqrt{d/h}}\right)\widetilde{\mathbf{V}}_i = \text{softmax}\left(\frac{\mathbf{Q}_i \cdot f(\mathbf{K}_i)^\top}{\sqrt{d/h}}\right) f(\mathbf{V}_i), \tag{3}$$

where h is the number of attention heads, and i is the head ID. The $\{\widetilde{\mathbf{S}}_1, \cdots, \widetilde{\mathbf{S}}_h\}$ are concatenated as the final $\widetilde{\mathbf{S}}$, which is the context-aware representation. We apply multiple layers of the above self-attention to facilitate in-depth fusion of the item and the context.

The complexity of our self-attention is reduced from $O(n^2)$ to $O(nk)$, as the item only needs to attend to a constant number of latent interests. Although many efficient Transformers [16] achieve the same complexity reduction, they require a fixed value of n, which is inflexible on varying sequence length. Some efficient Transformers also try to reduce the complexity from the perspective of hidden-dim (i.e., from $n \times d$ to $n \times k$) [2, 14]. However, the running cost will still be vulnerable to long sequences, and the acceleration is achieved with the potential loss of context representation quality.

Table 1: Statistics of the datasets after preprocessing.

Dataset	# Users	# Items	# Actions	# Avg.length	Sparsity
Yelp	56,590	75,159	2,290,516	40.47	99.94%
Books	19,214	60,707	1,733,934	90.24	99.85%
ML-1M	6,040	3,629	836,478	138.51	96.18%

2.2 Decoupled Position Encoding

Conventional SANs-based recommenders mix up item embeddings ($\mathbf{E}$) and position embeddings ($\mathbf{P}$) to introduce the sequential relations, i.e., $(\mathbf{E}+\mathbf{P})$. As a result, the relevance between two items (e.g., item i and item j) is $(\mathbf{E}_i+\mathbf{P}_i)(\mathbf{E}_j+\mathbf{P}_j)^\top$, which equals to $\mathbf{E}_i\mathbf{E}_j^\top+\mathbf{P}_i\mathbf{P}_j^\top+\mathbf{E}_i\mathbf{P}_j^\top+\mathbf{P}_i\mathbf{E}_j^\top$. However, the item-to-position correlations ($\mathbf{E}_i\mathbf{P}_j^\top$ and $\mathbf{P}_i\mathbf{E}_j^\top$) are not very strong, and they may limit the model's capability of capturing sequential relations from a sequence [7]. In this work, we propose decoupled position encoding to model sequential relations between items, which is independent of modeling context-aware representations:

$$\widetilde{\mathbf{S}} = \widetilde{\mathbf{A}}_{item} \cdot f(\mathbf{E}\mathbf{W}_V) + \widetilde{\mathbf{A}}_{pos} \cdot \mathbf{E}\mathbf{W}_V, \tag{4}$$

where f is from Eq. 2; $\widetilde{\mathbf{A}}_{item}$ is attention matrix calculated in Eq. 3: $\widetilde{\mathbf{A}}_{item} = \text{softmax}(\mathbf{Q}\cdot f(\mathbf{K})^\top/\sqrt{d/h})$, and $\mathbf{Q}=\mathbf{E}\mathbf{W}_Q, \mathbf{K}=\mathbf{E}\mathbf{W}_K$; $\widetilde{\mathbf{A}}_{pos}$ is calculated as: $\widetilde{\mathbf{A}}_{pos} = \text{softmax}(\mathbf{P}_Q\cdot(\mathbf{P}_K)^\top/\sqrt{d/h})$, and $\mathbf{P}_Q=\mathbf{P}\mathbf{U}_Q$, $\mathbf{P}_K=\mathbf{P}\mathbf{U}_K$ and $\mathbf{U}_Q, \mathbf{U}_K \in \mathbb{R}^{d\times d}$ are learnable parameters. According to the above equation, $\widetilde{\mathbf{A}}_{pos}$ is independently calculated parallel to $\widetilde{\mathbf{A}}_{item}$ (the item-to-interest interaction). By doing so, the sequential relations are explicitly specified without being affected by the item-to-position correlations, which improves the representation quality of our low-rank decomposed self-attention.

Besides, $\widetilde{\mathbf{A}}_{pos}$ can be computed once and shared across all the users within a common batch, as it is independent of the specific input. In other words, the computation cost of $\widetilde{\mathbf{A}}_{pos}$ is almost negligible for both training and testing stages.

2.3 Prediction Layer and Loss Function

Same as Transformer [15] to endow the model with nonlinearity, we apply a fully connected feed-forward network to $\widetilde{\mathbf{S}}^l$ in each self-attention layer l and obtain the result $\widetilde{\mathbf{F}}^l$. With the first t items encoded, the next item is predicted based on the last-layer's output $\widetilde{\mathbf{F}}_t^L$ of the t-th item (L is the number of self-attention layers). We use inner-product to measure user's preference to an arbitrary item i:

$$Pr(i|\{i_1^u,...,i_t^u\}) = \langle\widetilde{\mathbf{F}}_t^L, \mathbf{E}_i\rangle. \tag{5}$$

Finally, we adopt the cross-entropy loss to train our model:

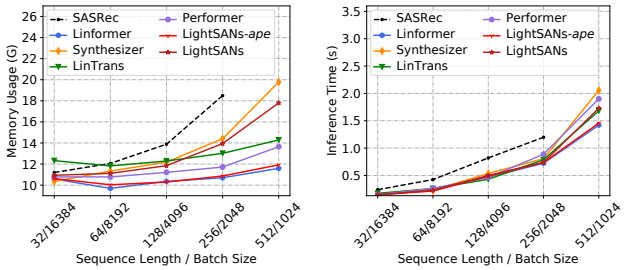
$$\mathcal{L} = -\log\frac{\exp(\langle\widetilde{\mathbf{F}}_t^L, \mathbf{E}_g\rangle)}{\sum_{i=1}^{|I|}\exp(\langle\widetilde{\mathbf{F}}_t^L, \mathbf{E}_i\rangle)}, \tag{6}$$

where item g is the ground truth item, $|I|$ is the number of all items.

3 EXPERIMENT

3.1 Experimental Settings

3.1.1 Datasets and Implementation Details. We use three real-world benchmark datasets, including Yelp, Amazon Books, and ML-1M, with their statistics shown in Table 1. Following previous works [5, 12], we apply the *leave-one-out* strategy for evaluation, and employ HIT@K and NDCG@K to evaluate the performance. For each user, we rank the ground truth item in the test set with all other items of the dataset. The models are implemented based on a popular open-source recommendation framework RecBole [17]. All codes, datasets and parameter settings are open-sourced[2].



(a) Memory Usage **(b) Inference Time**

Figure 2: Efficiency w.r.t Sequence Length on ML-1M dataset. SASRec is out of memory at sequence length = 512.

3.1.2 Baseline Models. Two kinds of baselines are considered, (1) *general recommendation methods*: Pop, FPMC [10], GRU4Rec [4], NARM [8], SASRec [5] and BERT4Rec [12]; (2) *efficient Transformers*: Synthesizer [14], LinTrans [6], Linformer [16] and Performer [2]. We mainly introduce the second kind of methods.

(1) **Linformer** reduces the length dimension of $\mathbf{K}$ and $\mathbf{V}$ from n to k through $n\times k$ linear mappings. However, the mapping components have to be re-trained given different sequence lengths, as the parameters of linear mappings require a fixed value of n. (2) **Synthesizer** leverages synthetic attention weights, which are factorizations of two randomly initialized low-rank matrices; **Performer** leverages Fast Attention via positive Orthogonal Random features approach (FAVOR+) to approximate the full-scale attention kernels. Both methods reduce the hidden-dim cardinality (from d to k), in contrast to reducing sequence length cardinality as LightSANs and Linformer (from n to k). (3) **Linear Transformer** (LinTrans) uses a kernel-based formulation of self-attention and the associative property of matrix products to calculate the attention weights. The complexity is changed from $O(n^2)$ to $O(nd)$, which means it only works for the cases where $n \gg d$. The time and space complexities of LightSANs ($O(nk)$) are the same as Linformer and Performer.

3.2 Main Results

3.2.1 Evaluation on Effectiveness. The overall performance is reported in Table 2, and we have the following observations. All SANs-based methods are better than other approaches because of the high-quality context-aware representations generated from the multi-head self-attention. LightSANs and LightSANs-*ape* (this variant replaces decoupled position encoding with the absolute position encoding in vanilla SANs [5] to evaluate the performance of our low-rank decomposed self-attention) yield more competitive results than other approaches across all evaluation metrics. Such results indicate that our proposed self-attention, highlighted by its item-to-interest interaction, generates more effective context-aware representation. Besides, LightSANs outperform LightSANs-*ape*, thanks to the decoupled position encoding.

[2]https://github.com/RUCAIBox/LightSANs

Table 2: Performance comparison (%) of all methods on three datasets. The best performance and the second best performance methods are denoted in bold and underlined fonts respectively.

Datasets	Metric	Pop	FPMC	GRU4Rec	NARM	SASRec	BERT4Rec	Synthesizer	LinTrans	Linformer	Performer	LightSANs-*ape*	LightSANs
Yelp	HIT@10	1.7	2.31	4.37	4.49	<u>5.09</u>	4.89	4.97	4.72	4.60	4.74	5.06	**5.48**
	NDCG@10	0.82	1.08	2.15	2.29	<u>2.76</u>	2.62	2.63	2.45	2.41	2.43	2.66	**2.89**
Books	HIT@10	3.95	7.97	8.08	8.16	8.43	8.10	8.16	8.61	8.12	<u>8.66</u>	8.22	**8.76**
	NDCG@10	1.56	3.98	4.02	4.12	4.14	4.03	4.05	4.20	4.03	<u>4.23</u>	4.06	**4.25**
ML-1M	HIT@10	8.15	12.32	21.30	21.75	22.11	21.99	21.40	<u>22.56</u>	14.66	18.73	22.37	**22.84**
	NDCG@10	4.04	5.89	10.91	10.98	11.21	10.99	10.84	<u>11.32</u>	7.28	10.10	**11.51**	**11.45**

Table 3: Comparison w.r.t #Parameters and GFLOPs.

Datasets	Metrics	SASRec	LightSANs-*ape*	LightSANs
Yelp	Params(M)	4.916	4.919 (1.00 times)	4.936 (1.00 times)
	GFLOPs	9.552	5.635 (0.58 times)	8.067 (0.84 times)
Books	Params(M)	3.995	3.998 (1.00 times)	4.015 (1.01 times)
	GFLOPs	18.005	9.183 (0.51 times)	14.663 (0.81 times)
ML-1M	Params(M)	0.345	0.350 (1.01 times)	0.367 (1.06 times)
	GFLOPs	28.910	13.219 (0.46 times)	22.968 (0.79 times)

Additional observations: GRU4Rec and NARM perform better than Pop and FPMC, thanks to the utilization of neural networks. NARM performs better than GRU4Rec, as it uses the attention mechanism to model the user's sequential behavior in each session.

3.2.2 Evaluation on Efficiency. The efficiency is compared between SASRec (the representative of original self-attention) and LightSANs. The computation cost is measured with gigabit floating-point operations (GFLOPs) on the self-attention module with position encoding; meanwhile, the model scale (measured with #Parameters) is also presented. As shown in Table 3, different models have almost the same amount of parameters, because the item embedding layer and feed-forward networks are the dominant components. Despite the similar model scales, significant acceleration is achieved by the proposed methods. Especially on ML-1M dataset with longer sequences than others, LightSANs-*ape* achieve more than 2× speed-up performance. LightSANs are slower than LightSANs-*ape*, this is because our decoupled position encoding needs extra computation cost to model user's sequential patterns. It is a trade-off between model's efficiency and additional performance gain. Meanwhile, LightSANs are still more efficient than the original self-attention.

Furthermore, we plot the memory usage and inference time of LightSANs and other SANs-based methods w.r.t. sequence length, while holding the total number of items fixed (all are tested on a Tesla P40 GPU). As shown in Figure 2, with the increase of sequence length, the cost of SASRec grows dramatically due to its quadratic complexity. Besides, for LightSANs-*ape*, LightSANs and other methods (LinTrans, Linformer, Performer), the memory usage retains relatively lower, and the inference speed is faster at long sequences. Although the model scale of LighSANs is slightly larger than SASRec, our approach decreases the complexity of attention matrix from $O(n^2)$ to $O(nk)$, which significantly reduces the memory cost.

3.3 Detailed Performance Analysis

We conduct effectiveness analysis of key designs in LightSANs, as shown in Table 4.

• **Effectiveness of decoupled position encoding.** The position encoding directly influences the modeling of items' sequential relations in SANs. Here we examine three types of position encoding applied to our self-attention. Firstly, without position embeddings, the performances drop a lot on both datasets, especially on ML-1M dataset with longer sequences. This means position information is essential to self-attention. Besides, both absolute (LightSANs–*ape*) [5] and relative [11] position encoding are worse than decoupled position encoding, which verifies the rationality of decoupling items' relevance and sequential relations.

Table 4: Effectiveness analysis about key componenets of LightSANs. {HIT, NDCG}@10 is adopted for evaluation.

Model		Yelp		ML-1M	
		HIT	NDCG	HIT	NDCG
Base Methods	SASRec	5.09	2.76	22.11	11.21
	LightSANs	**5.48**	**2.89**	**22.84**	**11.45**
Position	remove position	4.68	2.45	19.63	9.78
	absolute position	5.06	2.66	22.37	**11.51**
	relative position	5.08	2.70	22.01	11.06
Low-Rank Decomposition	remove low-rank	4.95	2.61	22.34	11.19
	SVD	3.36	1.69	13.29	6.74

• **Effectiveness of item-to-interest aggregation.** Since our item-to-interest aggregation is an essential component for modeling context-aware representation, we examine how it affects the final performance comparing with a simple decomposition method. We remove the low-rank decomposition part from LightSANs, where the performance becomes significantly lower. Besides, the accuracy drops on Yelp dataset while it performs similarly on ML-1M dataset compared with SASRec. For both cases, the acceleration achieved by the low-rank decomposition brings no negative effect to the prediction quality. We also apply singular value decomposition (SVD) to $\mathbf{K}$ and $\mathbf{V}$, choosing the item embeddings with larger singular values from the sequence. However, this simplification does not work as it is hard to be optimized in an end-to-end way.

4 CONCLUSION

In this paper, we proposed a Transformer variant LightSANs for next-item recommendation. Compared with orignal self-attention achitecture, LightSANs can learn context-aware representation for user history more effectively, and capture sequential relations between items more efficiently. Our approach can yield high-quality recommendation results with improved efficiency. Extensive experiments have shown that our approach outperforms a number of competitive baselines. As future work, we will test the model's performance on very long sequences, e.g., thousands of actions in each user history. Besides, we will also consider applying LightSANs to more scenarios beyond next-item prediction, e.g., pre-trained user models [18].

REFERENCES

[1] Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *CoRR* abs/2004.05150 (2020).

[2] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. 2020. Rethinking Attention with Performers. *CoRR* abs/2009.14794 (2020).

[3] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V. Le, Mark Z. Mao, Marc'Aurelio Ranzato, Andrew W. Senior, Paul A. Tucker, Ke Yang, and Andrew Y. Ng. 2012. Large Scale Distributed Deep Networks. In *NIPS*. 1232–1240.

[4] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR 2016*.

[5] W.-C. Kang and J. J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *ICDM 2018*. 197–206.

[6] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *ICML (Proceedings of Machine Learning Research)*, Vol. 119. PMLR, 5156–5165.

[7] Guolin Ke, Di He, and Tie-Yan Liu. 2020. Rethinking Positional Encoding in Language Pre-training. *CoRR* abs/2006.15595 (2020).

[8] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma. 2017. Neural Attentive Session-based Recommendation. In *CIKM 2017*. 1419–1428.

[9] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. 2019. Rethinking the Value of Network Pruning. In *ICLR (Poster)*. OpenReview.net.

[10] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW 2010*. 811–820.

[11] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-Attention with Relative Position Representations. In *NAACL-HLT (2)*. Association for Computational Linguistics, 464–468.

[12] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *CIKM 2019*. 1441–1450.

[13] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *WSDM*. ACM, 565–573.

[14] Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2020. Synthesizer: Rethinking Self-Attention in Transformer Models. *CoRR* abs/2005.00743 (2020).

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is All you Need. In *NeurIPS 2017*. 5998–6008.

[16] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-Attention with Linear Complexity. *CoRR* abs/2006.04768 (2020).

[17] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Kaiyuan Li, Yushuo Chen, Yujie Lu, Hui Wang, Changxin Tian, Xingyu Pan, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. 2020. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. *CoRR* abs/2011.01731 (2020).

[18] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *CIKM*. ACM, 1893–1902.