Locker: Locally Constrained Self-Attentive Sequential Recommendation

Zhankui He
UC San Diego
San Diego, USA
zhh004@eng.ucsd.edu

Handong Zhao
Adobe Research
San Jose, USA
hazhao@adobe.com

Zhaowen Wang
Adobe Research
San Jose, USA
zhawang@adobe.com

Zhe Lin
Adobe Research
San Jose, USA
zlin@adobe.com

Ajinkya Kale
Adobe Research
San Jose, USA
akale@adobe.com

Julian McAuley
UC San Diego
San Diego, USA
jmcauley@eng.ucsd.edu

ABSTRACT

Recently, self-attentive models have shown promise in sequential recommendation, given their potential to capture user long-term preferences and short-term dynamics simultaneously. Despite their success, we argue that self-attention modules, as a non-local operator, often fail to capture short-term user dynamics accurately due to a lack of inductive local bias. To examine our hypothesis, we conduct an analytical experiment on controlled *'short-term'* scenarios. We observe a significant performance gap between self-attentive recommenders with and without local constraints, which implies that short-term user dynamics are not sufficiently learned by existing self-attentive recommenders.

Motivated by this observation, we propose a simple framework, (Locker) for self-attentive recommenders in a plug-and-play fashion. By combining the proposed local encoders with existing global attention heads, Locker enhances *short-term* user dynamics modeling, while retaining the *long-term* semantics captured by standard self-attentive encoders. We investigate Locker with five different local methods, outperforming state-of-the-art self-attentive recommenders on three datasets by 17.19% (NDCG@20) on average.

CCS CONCEPTS

• **Information systems → Personalization**.

KEYWORDS

sequential recommendation, self-attentive recommender

ACM Reference Format:
Zhankui He, Handong Zhao, Zhaowen Wang, Zhe Lin, Ajinkya Kale, and Julian McAuley. 2021. Locker: Locally Constrained Self-Attentive Sequential Recommendation. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3459637.3482136

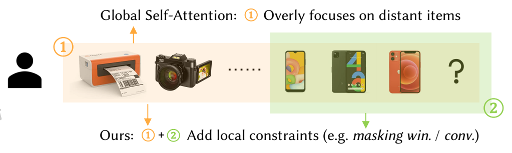

Global Self-Attention: ① Overly focuses on distant items
①
......
?
②
Ours: ①+② Add local constraints (e.g. *masking win.* / *conv.*)

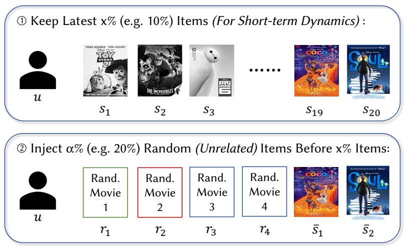
Figure 1: Global self-attention tends to overly focus on distant items without inductive local bias.

1 INTRODUCTION

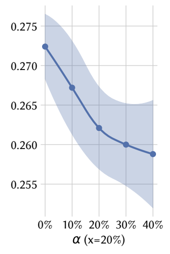
Sequential recommenders aim to balance *long-term* user preferences (e.g. preference toward action movies) with the *short-term* context of their recent actions (e.g. the last movie they watched). Considering both *long-* and *short-term* patterns simultaneously often improves recommendation accuracy, including Markov Chain (MC) based [10, 22] and RNN/CNN based approaches [5, 15, 24, 28].

To capture 'flexible order' (i.e., both long- and short-term preference) from sequential user data, self-attentive recommenders (SAR) have recently emerged as the state-of-the-art for both item-only [13, 17, 23, 26] and feature-rich [19, 30] sequential recommendation tasks, where self-attention [25] plays a central role by calculating item-to-item attention weights for the entire user behavior sequence. Some recent works [18, 26] improve self-attentive recommenders by introducing user models for long-term semantics, which implicitly assume self-attentive recommenders can handle short-term user dynamics well. However in this paper we argue that existing 'vanilla' self-attention (called *global self-attention* below) in self-attentive recommenders fails to sufficiently capture the importance of short-term user dynamics.

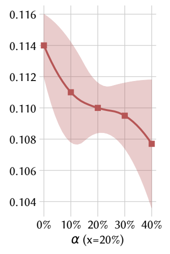
We first conduct a motivating experiment on a 'short-term' dataset. Though global self-attention could learn the correct semantics with *sufficient* data theoretically [29], our experimental results reveal that for real-world sequential recommendation tasks with limited data, global self-attention — as a non-local operator — tends to overly focus on distant historical items, resulting in performance degradation. An illustrative example from Figure 1 shows some distant items (e.g. printer, camera) are less related to user short-term interests (e.g. a mobile phone), while global self-attention is often insufficient to learn short-term dynamics accurately (and overly focuses on distant items in practice). Recent work in linguistics has shown that appropriate inductive local and other biases improve self-attention's generalization ability [6, 7, 16, 27]; this idea is yet to be widely adopted in the context of recommendation.



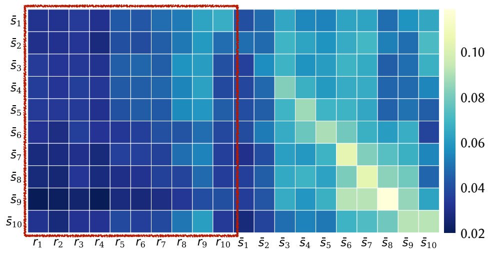
(a) Short-term data processing example.



(b) HR@20 *vs.* α



(c) N@20 *vs.* α



(d) Average attention map (len.=20, #sample=200).

Figure 2: Motivating experiments to show short-term user dynamics are not sufficiently learned my global self-attention.

In this work, we propose a simple framework, *<u>L</u>ocally <u>C</u>onstrained <u>S</u>elf-attentive <u>R</u>ecommender* (LOCKER), building on self-attentive networks. LOCKER enhances the ability to capture short-term user dynamics via local constraints (*local* encoder) while maintaining the capability to model long-term user preferences. For the local encoder, we investigate different local operators (e.g., model- or masking-based local encoders); for the global encoder, we adopt existing global self-attention networks. Experiments show that LOCKER with different local encoders outperforms existing self-attentive recommenders with small computational overhead.

2 PRELIMINARIES

2.1 Problem Setup

We are given a user set $\mathcal{U}$, an item set $\mathcal{I}$, and a set of user behavior sequences $\mathcal{S} = \{S_1, S_2, \ldots, S_{|\mathcal{U}|}\}$. Each user sequence consists of (chronologically ordered) item interactions $S_u = (s_1^{(u)}, s_2^{(u)}, \ldots, s_{N_u}^{(u)})$, where $S_u \in \mathcal{S}$, $u \in \mathcal{U}$, $s_i^{(u)} \in \mathcal{I}$ and N_u is the sequence length. Given the interaction history S_u, we predict the next item $s_{N_u+1}^{(u)}$.[1]

2.2 Self-Attentive Recommenders (SAR)

Self-attentive recommenders [13, 17, 23, 26, 30] rely heavily on *global self-attention*, though input types, training, or masking strategies vary. Global self-attention seeks to identify 'relevant' items from users' entire action sequences. Formally, $\mathrm{H}_i^l \in \mathbb{R}^{1\times d}$ is an embedding for s_i after the lth self-attention layer. A value vector $\tilde{V}_i$ from global multi-head (#heads=M) self-attention is calculated as:

$$\tilde{V}_i = [\tilde{V}_i^{(1)}; \ldots, \tilde{V}_i^{(m)}; \ldots; \tilde{V}_i^{(M)}]\mathrm{W}_O,$$
$$\text{where} \quad \tilde{V}_i^{(m)} = \sum_{j=1}^{N} f_{\mathrm{att}}\left(\mathrm{Q}_i^{(m)} \to \mathrm{K}_j^{(m)}\right) \cdot \mathrm{V}_j^{(m)}, \tag{1}$$

where f_{att} is an attention function (e.g. scaled dot-product attention [25]) to calculate the 'relevance' weight for any item-to-item pair (i.e. Q_i to K_j) in the input sequence; $\mathrm{W}_Q^{(m)}, \mathrm{W}_K^{(m)}, \mathrm{W}_V^{(m)} \in \mathbb{R}^{d\times d/M}$ are the m-th learnable projection matrices for input query $\mathrm{Q}_i^{(m)} = \mathrm{H}_i^l\mathrm{W}_Q^{(m)}$, key $\mathrm{K}_i^{(m)} = \mathrm{H}_i^l\mathrm{W}_K^{(m)}$ and value $\mathrm{V}_i^{(m)} = \mathrm{H}_i^l\mathrm{W}_V^{(m)}$; $\mathrm{W}_O \in \mathbb{R}^{d\times d}$ is a learnable projection matrix to get $\tilde{V}_i$ from concatenated vectors. Then models generate the next layer H_t^{l+1} using $\tilde{V}_i$ from Equation (1) with Residual Connections [9], `LayerNorm` [1] and `Pointwise Feed-Forward` Networks [25].

[1]Without loss of generality, we omit the user identifier u to simplify notation below.

2.3 SAR Needs Local Constraints

To show global self-attention is not sufficient for capturing short-term user dynamics in sequential recommendation, we design an analytical task on the widely used ML-1M dataset [8] (see dataset details in Section 4). We generate our training data from ML-1M following two steps (Figure 2a shows an example): (1) To focus on short-term dependencies, we truncate user sequences, retaining the last x% of items. (2) To investigate the ability of global attention to capture short-term dynmaics, we prepend α% random (non-meaningful) items before these x% items. Ideally, if global self-attention can capture 'flexible order' interactions from data, injected random items should be given little attention weight and will not harm recommendation performance significantly.

2.3.1 Model. We adopt BERT4Rec [23], a representative self-attentive recommender to fit the dataset. We set hidden size $d = 64$ with other optimal hyper-parameters via grid search. Other self-attentive recommenders [2, 13, 26, 30] use similar self-attention modules.

2.3.2 Evaluation. We follow [13, 23] to conduct a *leave-last-out* data split (i.e. for each sequence, using the first N-2 items for training, the (N-1)th for validation and the Nth for testing). We choose truncated Hit Ratio (HR@K) and Normalized Discounted Cumulative Gain (N@K) [13, 23] to measure ranking quality ($K = 20$). According to evaluation reviews [14], we use **all-item ranking**.

2.3.3 Observations. We evaluate recommendation performance with different random item length ratios α and observe:

- **Performance gap:** Figures 2b and 2c show that without inductive bias, global self-attention consistently performs significantly worse than the local model (i.e. $\alpha = 0$), even though theoretically sufficient training data helps global self-attention to capture short-term patterns correctly.
- **Behavior sequence length:** Figures 2b and 2c show that performance consistently drops with increasing α. Presumably, global self-attention considers all tokens, absorbing more noise from distant tokens as sequences get longer.
- **Attention map:** Figure 2d visualizes the average attention map. Global self-attention assigns larger weight to closer tokens ('brighter' neighbors) where the average attention score $\bar{f}_{\mathrm{att}}$ is 0.062. However, without local constraints, the

model still absorbs noise from distant unrelated items with average attention score $\bar{f}_{\mathtt{att}}$ score 0.038 (red box).

3 METHODOLOGY

Motivated by our preliminary experiment, we propose a generic framework, Locker, for self-attentive recommenders in a plug-and-play fashion, to enhance the ability to capture short-term dynamics, while maintaining the ability to capture long-term semantics. We then investigate different local encoders under Locker.

3.1 Locker Framework

Locker introduces local constraints into existing self-attention networks seamlessly. Like Equation (1), we concatenate output value vectors from M different attention heads, but Locker defines $M_l + M_g = M$ to split attention heads as M_l local encoders and M_g global encoders. Formally,

$$\tilde{V}_i = [\tilde{V}_{i,l}^{(1)}; \ldots; \tilde{V}_{i,l}^{(M_l)}; \tilde{V}_{i,g}^{(1)}; \ldots; \tilde{V}_{i,g}^{(M_g)}]\mathrm{W}_O, \tag{2}$$

where $\tilde{V}_{i,l}^{(m_l)}$ ($\tilde{V}_{i,g}^{(m_g)}$) is an output value vector from the local (global) encoder. We are interested in the role of explicit local encoders. So we simply keep global encoders as the same global attention head in Equation (1) and investigate several different local encoders, including model-based and masking-based encoders.

3.2 Model-based Local Encoder

For the model-based local encoder, we generate $\tilde{V}_{i,l}^{(m_l)}$ using neural-network operators with inductive local bias.

3.2.1 Fixed-Depth RNN (Locker+RNN). Recurrent networks are effective at short-term sequence modeling [11]. For our local encoder, to enhance the model's ability to capture short-term dynamics (while maintaining efficiency), it is natural to introduce a fixed-depth RNN module as a local encoder:

$$\tilde{V}_{i,l}^{(m_l)} = g\big(\mathrm{V}_{i,l}^{(m_l)},\ \underbrace{g(\mathrm{V}_{i-1,l}^{(m_l)},\ \ldots)}_{\text{recurrent depth } s}\big), \tag{3}$$

where g is the recurrent neural unit; here we choose Gated Recurrent Units (GRU) [3] as in GRU4Rec [11]. Here we use a GRU with fixed and small depth s to simplify computation and concentrate on short-term user dynamics.

3.2.2 Convolutional Network (Locker+Conv). Convolutional network is another option to model neighborhood dynamics. We define a convolution-based encoder for $\tilde{V}_{i,l}^{(m_l)}$ as:

$$\tilde{V}_{i,l}^{(m_l)} = [\mathrm{c}_1; \ldots; \mathrm{c}_{d/M}],\ c_j = \mathtt{act}\left(V_{[i]_s,l}^{(m_l)} \odot \mathrm{W}^{(j)}\right), \tag{4}$$

where $\odot$ denotes an inner product operator like [24], $\mathrm{V}_{[i]_s,l}^{(m_l)} \in \mathbb{R}^{s \times d/M}$ denotes the local [i-(s-1)/2,...,i+(s-1)/2] rows (size s is odd number) in $\mathrm{V}_l^{(m_l)} \in \mathbb{R}^{N \times d/M}$. $W^{(j)} \in \mathbb{R}^{s \times d/M}$ denotes the j-th convolutional kernel. act is an activation function to introduce non-linearity such as ReLU. Compared to CASER [24], which used convolutional networks to capture point-level and union-level item similarities, we adopt convolutional networks as a local operator to enhance short-term user dynamics modeling.

3.3 Masking-based Local Encoder

For a masking-based local encoder, we reconsider the global attention function $f_{\mathtt{att}}$ by introducing locality-aware masking to enhance the ability to capture short-term dynamics. For Equation (1), where $f_{\mathtt{att}}$ in detail is defined by 'relevance' logit w_{ij} with position i, j and fed into the softmax layer for normalization, i.e.:

$$f_{\mathtt{att},l}(\mathrm{Q}_i \to \mathrm{K}_j) = \frac{\exp(w_{ij}) \cdot \sigma_{ij}}{\sum_{k=1}^{N} \exp(w_{ik}) \cdot \sigma_{ik}} \tag{5}$$

where masking score $\sigma_{ij} \equiv 1$ for global self-attention. In the masking-based local encoder, we enhance the ability to capture short-term dynamics by changing the masking score σ_{ij} with different strategies.

3.3.1 Fixed Window (Locker+Win). Fixed window simply deactivates all distant tokens, where σ_{ij} is defined as:

$$\sigma_{ij} = \mathbb{I}(|i-j| \le s), \tag{6}$$

where $\mathbb{I}$ is an indicator function. Therefore, the attention map is masked by a fixed-length window to deactivate the dependency on distant (distance > s) tokens.

3.3.2 Gaussian Initialization (Locker+Initial). Locker+Win. predefines 'hard' and 'static' masking scores for all training data, which could be too rigid. We seek to introduce 'trainable' masking scores with good initialization, which is one way to introduce a locality prior into this encoder. The masking operation $\exp(w_{ij}) \cdot \sigma_{ij}$ in Equation (5) can be rewritten as $\exp(w_{ij} + \ln \sigma_{ij})$. We seek to 'learn' the unbounded adjustable weight $p_{i-j} = \ln \sigma_{ij}$, where $i - j$ means we map different distance to a trainable weight p_{i-j}.

We can perform weight initialization following a Gaussian-like function (e.g., $p_{i-j}^0 = a\exp(-(i-j)^2/b)$), where local concentration exists in the neighborhood. Note that changing weight initialization does not guarantee an explicit local 'fixed window' exists after training, but the initialization bias encourages the model to capture local patterns from a better starting point than (e.g.) uniform initialization and can adjust during training. To further encourage trainable weights to capture locality from data, we remove the positional embeddings (in standard SAR [13, 23]) for local encoder vectors, i.e., only incorporating positional embeddings into the global encoder key and query vectors.

3.3.3 Adaptive Predictor (Locker+Adapt). Locker+Initial adjusts soft scores but cannot encode additional information, such as a user identifier u. We further extend Locker+Initial to a parameterized adaptive predictor pred to predict different masking scores, i.e.:

$$p_{i-j}^{(u)} = \mathtt{pred}\left(\mathrm{V}_{i,l}^{(m_l)} + \mathrm{V}_{j,l}^{(m_l)} + v_u + b_{i-j}\right), \tag{7}$$

where we can additionally encode user information $v_u \in \mathbb{R}^{1 \times d}$, distance embedding $b_{i-j} \in \mathbb{R}^{1 \times d}$, and current value vectors $\mathrm{V}_{i,l}^{(m_l)}$ and $\mathrm{V}_{j,l}^{(m_l)}$. We construct user representations v_u following FISM [12] and FISSA [18] without extra user embeddings. Like Locker+Initial, we remove positional embeddings for local encoder vectors to encourage the model to learn locality from data. pred is a two-layer MLP model to learn more flexible masking scores with user, distance and current token information.

		Baseline Models					Locker *(Backbone: BERT4Rec)*					Improvement	
Dataset	Metric	PopRec	BPR-MF	SASRec	BERT4Rec	SSE-PT	+RNN	+Conv	+Win	+Initial	+Adapt	Avg.	Max.
Beauty	N@20	0.0048	0.0172	0.0206	0.0238	0.0232	0.0258	0.0297	0.0296	0.0303	**0.0311**	+23.11%	+30.67%
	HR@20	0.0131	0.0425	0.0496	0.0541	0.0547	0.0568	0.0661	0.0641	0.0652	**0.0672**	+16.78%	+22.85%
Clothing	N@20	0.0021	0.0035	0.0052	0.0062	0.0059	0.0070	0.0078	0.0074	0.0077	**0.0079**	+21.94%	+27.42%
	HR@20	0.0056	0.0089	0.0140	0.0153	0.0149	0.0166	0.0184	0.0174	0.0184	**0.0187**	+16.99%	+22.22%
ML-1M	N@20	0.0260	0.0498	0.1625	0.1783	0.1763	0.1930	**0.1980**	0.1831	0.1863	0.1893	+06.53%	+11.05%
	HR@20	0.0686	0.1298	0.3652	0.3870	0.3841	0.4012	**0.4119**	0.3972	0.3900	0.4047	+03.62%	+06.43%

Table 1: Model comparision. We tune Locker with M={2,4} where M_l={1,...,M-1}, and use similar settings for the backbone. Compared to the best baselines (underline), Avg. (Max.) for average (maximum) relative improvement of five local encoders.

	#Interaction	#Item	#Sequence	Average Length	Density
Beauty	353,962	54,542	40,226	8.80	1e-4
Clothing	831,816	162,193	108,489	7.67	1e-5
ML-1M	1,000,000	3,416	6,040	165.56	1e-2

Table 2: Data statistics.

4 EXPERIMENTS

4.1 Experimental Setting

4.1.1 Data. We consider the following datasets from different domains with various data distributions (see Table 2): **Beauty, Clothing** are datasets collected from Amazon in [20]. **ML-1M** [8] is a popular benchmark dataset for top-N recommendation. We follow the data pre-processing and splitting from [23] (details in Section 2.3.2).

4.1.2 Baselines. **PopRec**, A baseline recommending items according to item occurrences in the dataset. **BPR-MF** [21] A classic personalized ranking learning algorithm based on matrix factorization. **SASRec** [13]: A seminal method using self-attention mechanism for sequential recommendation. **BERT4Rec** [23]: A BERT-like [4] model capturing bi-directional contextual item information via a *cloze* task for next-item recommendation. **SSE-PT** [26]: A state-of-the-art self-attentive recommender, extends SASRec by introducing explicit user representations.

4.1.3 Evaluation and Implementation. We reuse the evaluation protocols in Section 2.3.2. We implement all models via PyTorch and use grid search with the same granularity to tune baselines (following each baseline’s suggestions). Locker uses the same training and hyper-parameter searching strategy as our backbone (BERT4Rec)[2].

4.2 Result Analysis

4.2.1 General Performance. Table 1 shows model ranking performance on three datasets. (1) Self-attentive sequential recommenders (SASRec, BERT4Rec, SSE-PT) consistently outperform classic methods by effectively leveraging sequential information with global self-attention networks. BERT4Rec outperforms SASRec by using bidirectional training. SSE-PT outperforms SASRec by introducing explicit user representations. (2) Our Locker framework outperforms all baselines consistently. Compared to the strongest global self-attention-based recommender (BERT4Rec, SSE-PT), our model gains about 17.19% N@20 and about 12.46% HR@20 improvements on average with a comparable number of parameters. Furthermore Locker with the most effective local encoder gains about 23.04% N@20 and about 17.67% HR@20 improvements on these three datasets. This shows the effectiveness of introducing inductive local bias into self-attentive recommenders with different local encoders.

[2]Locker implementation in https://github.com/AaronHeee/LOCKER

4.2.2 Local Encoder Discussion. For local encoders, (1) model- and masking-based encoders all outperform pure global self-attentive sequential recommenders. Interestingly, on ML-1M (with the longest average user sequences), model-based encoders exceed all (2) Presumably because RNN encodes actions ‘one-by-one’ and cannot capture more flexible ‘skip’ behaviors like other encoders, RNN performs the worst on these three datasets. (3) Conv. and Win. are both fixed-size encoders and can capture flexible item dependencies. The superiority of Conv. on three datasets may owe to introducing extra model parameters. (4) Initial. and Adapt. perform better than fixed-window Win. with learnable masking, where Adapt can encode more information (user, current tokens) thus further improves performance (though with longer training time, shown below). The characteristics of five encoders are summarized in Table 3.

Local Encoder	Skipped Behavior?	No Extra Param.?	Adapt. Size?
RNN			
Conv	✓		
Win	✓	✓	
Init./Adapt	✓		✓

Table 3: Multiple local encoder characteristics.

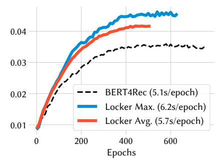


Table 4: NDCG@20 validation (Beauty).

4.2.3 Efficiency and Convergence. Table 4 records NDCG@20 curves on the Beauty validation set on a single Nvidia 2080s GPU. *Locker Max.* indicates Locker+Adapt with the slowest training speed, *Locker Avg.* shows the average NDCG@20 scores and training time of these five Locker models. Compared with BERT4Rec, we observe that (1) Our models achieve comparable performance with far fewer training epochs (~200 *versus* ~500). (2) Our models converge using similar training epochs with small computational overhead (5.7s/epoch on average *versus* 5.1s/epoch).

5 CONCLUSION

Self-attentive recommenders have shown promise in sequential recommendation, where global self-attention plays a critical role. In this paper, we find that—without any inductive bias—global self-attention cannot easily capture short-term user dynamics. Thus, we propose a framework, Locker, to introduce local inductive bias. We extend existing self-attention networks using five local encoders to enhance short term dynamics modeling and show the effectiveness of this idea on several datasets. In the future, more sophisticated inductive biases can be considered in self-attentive recommenders.

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).

[2] Xusong Chen, Dong Liu, Chenyi Lei, Rui Li, Zheng-Jun Zha, and Zhiwei Xiong. 2019. BERT4SessRec: Content-Based Video Relevance Prediction with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 27th ACM International Conference on Multimedia*. 2597–2601.

[3] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. 103–111.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.

[5] Robin Devooght and Hugues Bersini. 2017. Long and short-term recommendations with recurrent neural networks. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. 13–21.

[6] Maosheng Guo, Yu Zhang, and Ting Liu. 2019. Gaussian transformer: a lightweight approach for natural language inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6489–6496.

[7] Qipeng Guo, Xipeng Qiu, Pengfei Liu, Xiangyang Xue, and Zheng Zhang. 2020. Multi-Scale Self-Attention for Text Classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 7847–7854.

[8] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[10] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 191–200.

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

[12] Santosh Kabbur, Xia Ning, and George Karypis. 2013. Fism: factored item similarity models for top-n recommender systems. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 659–667.

[13] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[14] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1748–1757.

[15] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1419–1428.

[16] Jian Li, Zhaopeng Tu, Baosong Yang, Michael R Lyu, and Tong Zhang. 2018. Multi-Head Attention with Disagreement Regularization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2897–2903.

[17] Shihao Li, Dekun Yang, and Bufeng Zhang. 2020. MRIF: Multi-resolution Interest Fusion for Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1765–1768.

[18] Jing Lin, Weike Pan, and Zhong Ming. 2020. FISSA: fusing item similarity models with self-attention networks for sequential recommendation. In *Fourteenth ACM Conference on Recommender Systems*. 130–139.

[19] Chang Liu, Xiaoguang Li, Guohao Cai, Zhenhua Dong, Hong Zhu, and Lifeng Shang. 2021. Noninvasive Self-attention for Side Information Fusion in Sequential Recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4249–4256.

[20] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

[21] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. 452–461.

[22] 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*. 811–820.

[23] 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 *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1441–1450.

[24] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 565–573.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[26] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2020. SSE-PT: Sequential Recommendation Via Personalized Transformer. In *Fourteenth ACM Conference on Recommender Systems* (Virtual Event, Brazil) *(RecSys '20)*. Association for Computing Machinery, New York, NY, USA, 328–337. https://doi.org/10.1145/3383313.3412258

[27] Baosong Yang, Zhaopeng Tu, Derek F Wong, Fandong Meng, Lidia S Chao, and Tong Zhang. 2018. Modeling Localness for Self-Attention Networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4449–4458.

[28] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 582–590.

[29] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. 2020. Are Transformers universal approximators of sequence-to-sequence functions?. In *International Conference on Learning Representations*. https://openreview.net/forum?id=ByxRM0Ntvr

[30] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation.. In *IJCAI*. 4320–4326.