Knowledge-enhanced Session-based Recommendation with Temporal Transformer

Rongzhi Zhang[1],Yulong Gu[1],Xiaoyu Shen[2], Hui Su[3]
zhangrzpanda@gmail.com,guyulongcs@gmail.com
[1]Alibaba Group, China
[2]Saarland Informatics Campus, Germany
[3]Pattern Recognition Center, Tencent, China

ABSTRACT

Recent research has achieved impressive progress in the session-based recommendation. However, information such as item knowledge and click time interval, which could be potentially utilized to improve the performance, remains largely unexploited. In this paper, we propose a framework called **K**nowledge-enhanced **S**ession-based Recommendation with **T**emporal **T**ransformer (KSTT) to incorporate such information when learning the item and session embeddings. Specifically, a knowledge graph, which models contexts among items within a session and their corresponding attributes, is proposed to obtain item embeddings through graph representation learning. We introduce time interval embedding to represent the time pattern between the item that needs to be predicted and historical click, and use it to replace the position embedding in the original transformer (called temporal transformer). The item embeddings in a session are passed through the temporal transformer network to get the session embedding, based on which the final recommendation is made. Extensive experiments demonstrate that our model outperforms state-of-the-art baselines on four benchmark datasets.

KEYWORDS

session-based recommendation, temporal transformer, knowledge graph

ACM Reference Format:
Rongzhi Zhang[1],Yulong Gu[1],Xiaoyu Shen[2], Hui Su[3]. 2021. Knowledge-enhanced Session-based Recommendation with Temporal Transformer. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Session-based recommendation is a classic problem in recommendation systems [5]. It aims to predict the next items users will choose based on their historical behaviors *within the current session*. Recently, many researchers have used the idea of Natural Language Processing(NLP) task(like [1–3, 12, 16, 17, 28]) for reference to provide new ideas for session-based recommendation [21, 24, 27].

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

For session-based recommendation, the model architecture usually consists of two key components: (1) **Item Encoder** that projects each item into a low dimensional vector. In doing so, Smirnova and Vasile [18] concatenated the one-hot embedding of the item and its context representation into recurrent neural networks (RNNs). Wu et al. [23] further considered the temporal relationships of items. (2) **Session Encoder** which encapsulates all items within the current session into a compact vector. Hidasi et al. [4] proposed a session embedding model based on Gated Recurrent Units (GRUs). Li et al. [8] and Liu et al. [11] used attention mechanisms to capture users' local and global preferences. Recently, Xu et al. [25] further improved the recommendation performance by utilizing self-attention networks.

Despite the impressive success, two limitations exist in current models. (1) Item attributes involve important knowledge, and the context information of the item can be obtained by modeling them, which is beneficial to item recommendations. However, existing approaches ignore them when building the item encoder. (2) The session embedding ignores the time interval between behaviors. The time interval between two adjacent clicks in the user's click history is varied. When the time interval is different, the product that the user clicks next should also have a different relationship with the previous behavior. For example, if the user is currently buying beer, the click within 3 minutes may be a beer of a different brand, but after 10 minutes, they may be browsing products such as beer glasses. Existing self-attention networks like Transformers Vaswani et al. [22] are powerful at modelling sequential data with position embedding, but they fail to encode the time interval information for predictions, resulting in unsatisfactory results in session-based recommendations.

In this paper, we propose a Knowledge-enhanced Session-based Recommendation with Temporal Transformer (KSTT) to improve the performance of session-based recommendations. In comparison with state-of-the-art methods, our KSTT is equipped with two mechanisms to address the above-mentioned limitations: (1) We build a knowledge graph based on the attributes of the items and the order of clicks between the items, and use the graph neural network (GNN) to obtain the global context information of each node. (2) We use a well-designed mapping function to embed the timestamp into k-dimension vectors for modeling the time and time interval information. The mapping function is based on recent researches [6, 26], which replaces position embedding with time embedding to reveal more time patterns and recurrent attention. In summary, our contributions are listed as follows.

- We introduce the knowledge graph into session-based item recommendation, and use GNN to integrate the information of adjacent nodes in the graph.
- We introduce temporal transformer into session-based recommendation and testify its effectiveness.
- We conduct extensive experiments on four benchmarks datasets to demonstrate that KSTT can significantly overcome state-of-the-art baselines.

2 KSTT MODEL

In this paper, we propose a framework called Knowledge-enhanced Session-based Recommendation with Temporal Transformer (KSTT) for Session-based Recommendation. Given the training set, it has items $\mathcal{V} = \{v_1, v_2, ..., v_M\}$ and attributes $\mathcal{A} = \{a_1, a_2, ..., a_K\}$. Session $s_i = \{v_{i,1}, v_{i,2}, ..., v_{i,t}\}$ means that the i-th session contains a transaction with item $v_{i,t}$ at time t. The architecture of KSTT is demonstrated in Figure 1. Firstly, the Item encoder in KSTT learns the embeddings of items using Knowledge-enhanced Graph Neural Network. The nodes in the knowledge graph contains two types: items(v) and items' attributes(a). The edges in the graph represent the relations(r) between nodes. Secondly, the Session encoder in KSTT exploits Temporal Transformer using time encoder ϕ to learn session representation. Finally, KSTT calculates the prediction scores $\hat{y}$ for the next item.

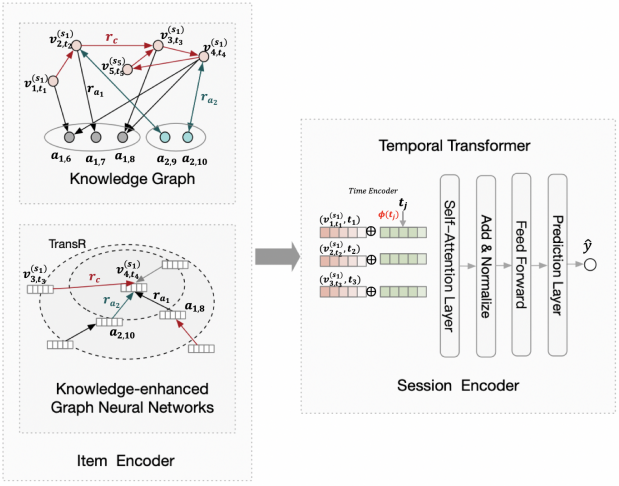

Figure 1: Architecture of KSTT.

2.1 Item Encoder with Knowledge-enhanced Graph Neural Network

The item encoder uses Knowledge-enhanced Graph Neural Network to learn item embeddings based on the knowledge graph.

Knowledge Graph Construction Previous methods [20, 23, 25] usually construct the session graph based on the sequential relationship between items. In this work, we propose the idea of constructing a directed *KG* based on the relationships in the session as well as the knowledge (e.g. attributes) of items. We construct the *KG*, where the nodes are items and these items' related attributes and the edges are the relationships between the nodes including sequential edges and semantic edges. The sequential edges are denoted as $(v_{i,t}, r_c, v_{i,t+1}), 1 \leq t < m$ and c is a fixed number. r_c is the sequential relationship of users' behaviors, and each edge represents that the user clicks the item $v_{i,t+1}$ after consuming item $v_{i,t}$. The semantic edges are denoted as $(v_{i,t}, r_j, a_{kj}), 0 < j \leq k+1$ and 0 < k < K which represent item $v_{i,t}$ and attribute a_{kj} are connected with relation r_j.

Knowledge-enhanced Graph Neural Network We exploit our proposed Knowledge-enhanced Graph Neural Network to learn the embeddings of items in the *KG*. We perform the knowledge graph embedding method TransR [10] to learn the semantic information and exploit Graph Convolutional Neural Networks [7] to aggregate information from neighbors to capture context information between nodes. We use the *(head,relation,tail)* entity triplets to learn the semantic representation of nodes. For each triplet, the embeddings of the head entity h, relation r and tail entity t are $d$dimension vector updated by graph convolutional neural network [7]. The update function of nodes in one graph convolutional layer can be represented as: $e'_{h/t} = norm(LeakyReLU(GEW))$, where $e_{h/t}$ is the representation of entity h or t. E is the embeddings for all entities and G is the adjacent matrix of knowledge graph. Graph convolution is computed through gathering the response from each neighbour, and multiple graph convolutional layers are stacked to extract high-level representation for nodes. Therefore, graph convolution is suitable for exchanging information in neighborhoods.

2.2 Session Encoder with Temporal Transformer

The Session Encoder exploits Temporal Transformer to capture the relationships among items in a session, and learn the representation of the session.

Temporal Transformer In the Transformer architectures [22], position embeddings are used to model the position information in one sequence. However, for session-based recommendation, this method only models the sequential order instead of time interval information between behaviors. In this work, we proposed Temporal Transformer to deal with this issue. Firstly, we encode each behavior in the session. For each historical behavior, we calculate the corresponding time interval, which is difference of timestamp between this behavior and prediction time. Then, we exploit the time encoder to encode the time interval into low dimensional vector. The embedding of the behavior v_i is defined as $E(v_i)$, computed by the sum of item embedding $IE_d(v_i)$ and time embedding $TE_d(v_i)$. Secondly, as Transformer did [22], we exploit Multi-head Self-Attention Networks and Feed Forward neural networks(FFN) to learn the representation of the session. The calculation method is defined as $L^{(l)} = \text{FNN}(\text{MultiHead}(L^{(l-1)}, L^{(l-1)}, L^{(l-1)}))$, where $MultiHead(\cdot,\cdot,\cdot)$ is a multi-head self-attentive network that takes a query matrix, a key matrix, and a value matrix as input. $L^{(0)}$ is initialized session representation including behavior embeddings $E(v_i)$ of this session.

Time Encoder We investigate three time embedding methods as time encoder: Time Bucket embedding method, Time2vec [6] and Mercer time embedding method [26].

Time Bucket Embedding method(TBE). This method uses log and floor function to discretize time into one-hot representation.

$$\Phi_{bucket}(t) = \text{One-hot}(\lfloor \log_2(\hat{t} - t) \rfloor) \tag{1}$$

Time2vec(T2v). T2v is designed to capture both periodic and non-periodic patterns in time [6]. The embedding is calculated using following equation

$$\Phi^i_{T2V}(t) = \begin{cases} w_i t + b_i, & if\ i = 1 \\ sin(w_i t + b_i), & if\ 1 \le i < d \end{cases} \tag{2}$$

where i is the dimension.

Mercer Time Embedding(MTE). Mercer Time Embedding(MTE) learns time representation with transition-invariant property [26]. The detailed Mercer time embedding function is given as follows:

$$\begin{aligned} \Phi_{MTE}(t) &= [\Phi_{w_1,d}(t), \Phi_{w_2,d}(t), ..., \Phi_{w_k,d}(t)]^T \\ \Phi_{w,d}(t) &= [\sqrt{c_1}, \sqrt{c_{2j}} cos(\frac{j\pi t}{\omega}), \sqrt{c_{2j+1}} sin(\frac{j\pi t}{\omega}), ...] \end{aligned} \tag{3}$$

where c and ω represent the parameters. ω should be initialized by uniformly distribution and ω_0 should be initialized as 0.

In summary, these time embedding methods capture different time patterns within one session which helps attention mechanism to learn session embedding.

2.3 Prediction and Optimization

We adopt the two phases with different loss functions to optimize KSTT. In one phase, we mainly optimize the loss function based on Lin et al. [10]. After obtaining entity embeddings from knowledge graph, we also utilize a projection matrix $\mathbf{M}_r$ to project the head entities h and tail entities t from the entity space to the relation space, represented as $\mathbf{M}_r\mathbf{h}$ and $\mathbf{M}_r\mathbf{t}$ respectively. Then we try to minimize the difference between $\mathbf{M}_r\mathbf{h} + \mathbf{r}$ and $\mathbf{M}_r\mathbf{t}$, formulated as the score function $g_r(h,t) = \|\mathbf{M}_r\mathbf{h} + \mathbf{r} - \mathbf{M}_r\mathbf{t}\|_2^2$. A lower score means that the triplet is more likely to be true. We sample head entities from all training nodes with difference relation and obtain a batch containing positive and negative pairs. We minimize the scores of negative pairs and maximize that of positive pairs for learning knowledge graph embeddings as follows.

$$\mathcal{L}_{KG} = \sum_{(h,r,t,t') \in T_{KG}} \ln \sigma \left[g_r(h,t) - g_r(h,t') \right] \tag{4}$$

where $T_{KG} = \{(h,r,t,t') \mid (h,r,t) \in \mathcal{G}, (h,r,t') \notin \mathcal{G}\}$, (h,r,t') is a random sampled negative triplet by replacing t with another entity t' in the knowledge base, and $\sigma(\cdot)$ is the sigmoid function.

In the other phase, we aim to predict the next items users will buy. First, we obtain the session representation s for each user from temporal transformer. Then we can compute the score $\hat{z}_i$ for each item v_i by $\hat{z} = s_h^\top v$. we apply a softmax function over $\hat{z}$ and obtain the final output vector $\hat{\mathbf{y}}$ which is the probability distribution of the items. we optimize KSTT with the cross-entropy of the prediction $\hat{\mathbf{y}}$ and the ground truth $\mathbf{y}$.

$$\mathcal{L}_{rec} = -\sum_{i=1}^{n} \mathbf{y}_i \log(\hat{\mathbf{y}}_i) + (1 - \mathbf{y}_i) \log(1 - \hat{\mathbf{y}}_i) \tag{5}$$

where n is the number of training data.

We jointly optimize the session-based recommendation and knowledge graph embedding by minimizing the following loss function: $\mathcal{L}_{KSTT} = \mathcal{L}_{rec} + \mathcal{L}_{KG} + \lambda ||\Theta||_2^2$, where Θ is the parameters of our model, and λ is the coefficients of the L_2 regularization.

3 EXPERIMENTS

3.1 Experiment setting

Datasets We choose three representative benchmark Yoochoose, Diginetica and Last-fm, to evaluate our model. **Yoochoose** is used as a challenge dataset for RecSys Challenge 2015. It records the click-streams from an E-commerce website within 6 months. As previous did, we split the session sequences by fixed cutoff to generate the train and test datasets, and then further split the train data by the most recent partition 1/4 and 1/64. **Diginetica** is used as a challenge dataset for CIKM Cup 2016. It contains the transaction data with the basic information of the products and transaction. **Last-fm** is a music dataset collected from the Last.fm API, which has been widely applied for music artist recommendations. After preprocessing, we keep the top 40,000 most popular artists and filter out sessions that are longer than 50 or shorter than 2 items.

Baselines In order to valid the effectiveness of our proposed KSTT model, we compare KSTT with the following representative methods: **POP** always recommends the most popular items in the whole training set. **S-POP** always recommends the most popular items in the current session. **Item-KNN** computes the similarity between each pair of items with their cosine distance in sessions. **BPR-MF** [14] calculates a pairwise ranking loss with a BPR objective function. **FPMC** [15] is a hybrid model for the next-basket recommendation based on the personalized transition matrix tube. **GRU4REC** [4] stacks multiple GRU layers to encode the session sequence into a latent vector, where a ranking loss is used to train the model. **NARM** [8] incorporates an attentive network to combine states of RNN. **STAMP** [11] uses attention layers to replace all RNN encoders in previous work. **RepeatNet** [13] proposes an encoder-decoder architecture to solve the reputation consumption problem. **SR-GNN** [23] applies a gated graph convolutional layer [9] to obtain item embeddings. **GC-SAN** [25] introduces a self-attention network to the session encoder.

Parameters and Metric Measure The item embedding size and the dropout ratio are set to 100 and 0.1 in all the methods. We use Adam as our optimizing algorithm and set the learning rate as 0.001. The batch size is set to 246. To speed up the converge in the training process, we set the mini-batch size as 256 in session graph and 512 in the knowledge graph.

For the evaluation metrics, we use R@K (Recall@K) and MRR@K (Mean Reciprocal Rank@K) to measure the recommendation performance.

3.2 Performance Comparison

Table 1 lists the performance of the proposed model and other state-of-art session-based recommendation methods. All of the methods fall into three categories: classical methods, DNN-based methods and GNN-based methods. Considering POP and S-POP, they ignore the complex relationship between items and make recommendation based on occurrence number. Even so, S-POP still outperforms other methods demonstrating the importance of session contextual information. Item-KNN achieves better results than FPMC. Item-KNN

Table 1: Performance comparisons on benchmark datasets.

	Method	*Diginetica*		*Yoochoose1/64*		*Yoochoose1/4*		*Last-fm*	
		R@20	MRR@20	R@20	MRR@20	R@20	MRR@20	R@20	MRR@20
Classical	POP	0.89	0.20	6.71	1.65	1.33	0.30	5.26	1.26
	S-POP	21.06	13.68	30.44	18.35	27.08	17.75	22.59	8.71
	Item-KNN	35.75	11.57	51.60	21.81	52.31	21.70	14.84	4.85
	BPR-MF	5.24	1.98	31.31	12.08	3.40	1.57	14.05	5.01
	FPMC	26.53	6.95	45.62	15.01	51.86	17.50	17.68	4.99
DNN-based	GRU4REC	29.45	8.33	60.64	22.89	59.53	22.60	17.90	5.39
	NARM	49.70	16.17	68.32	28.63	69.73	29.23	29.94	10.85
	STAMP	45.64	14.32	68.74	29.67	70.44	30.00	29.24	11.33
	RepeatNet	47.79	17.66	69.13	30.24	70.71	31.03	32.28	12.03
GNN-based	SR-GNN	50.73	17.59	70.57	30.94	71.36	31.89	30.75	12.53
	GC-SAN	52.48	18.20	71.20	30.48	71.46	31.47	32.46	12.22
	KSTT	**53.25**	**18.29**	**72.25**	**31.05**	**72.35**	**32.30**	**32.63**	**12.90**

utilizes the similarity between items without considering sequential information but MC-based methods often consider adjacent information. These demonstrate that inter-session information is more important than information of neighbors.

In particular, DNN-based methods, explicitly model the users' global behavioral preferences by local and global representation leading to superior performance against traditional methods. GNN-based methods are state-of-the-arts. They model the transactions within a session as a graph, which outperforms other methods significantly. Our proposed method KSTT is more powerful to model behaviors in the session and achieves the best performance on all four datasets in terms of R@20 and MRR@20, especially for *Yoochoose* dataset which most have 1.05% improvement for R@20 and 0.83% for MRR@20 .

3.3 Ablation Study

In this subsection, we conduct the ablation studies to verify the effectiveness of Knowledge-enhanced Graph Neural Network (KGNN) and Temporal Transformer (TT) combining Self-Attention Neural Networks (SAN) and Time Encoder (TE). We conduct the experiments on *Yoochoose1/64* and *Diginetica* and choose SRGNN as baseline. The experimental results are listed in Table 2, where + means **adding** this module or **replacing** the original similar module on the baseline. We can draw the following conclusions: (1) ***For KGNN***: KGNN module indeed helps GNN achieve better performance, evidencing that +KGNN overcomes SRGNN that only uses a session graph. In particular, the KGNN module brings 1.44 (50.73 to 52.17) improvement in R@20 and 0.39 (17.59 to 17.88) gains in MRR@20 on *Diginetica* and the improvement on *Yoochoose1/64* is 0.26 and 0.17, respectively. Since *Diginetica* has more attributes than *Yoochoose*, the former has more knowledge than the latter. Therefore, we can conclude that more knowledge information can bring better improvement. (2) ***For TT***: We use three time embedding functions. The results show that all the time representation methods help session encoder with more information, especially TBE. This demonstrates that TBE has superiority the inter-session time pattern for session-based recommendation.

In summary, our proposed modules can significantly improve the baseline model, which result in the state-of-the-art performance.

Table 2: Ablation study of modules.

Models	*Yoochoose1/64*		*Diginetica*	
	R@20	MRR@20	R@20	MRR@20
Baseline	70.48	30.81	50.73	17.59
Baseline (+KGNN)	70.74	30.98	52.17	17.88
Baseline (+SAN)	71.20	30.48	52.48	18.20
Baseline (+KGNN, +SAN)	71.53	30.50	53.10	18.25
Baseline (+KGNN, +SAN, +MTE)	72.09	30.63	53.12	18.24
Baseline (+KGNN, +SAN, +T2v)	72.10	30.70	53.15	18.27
Baseline (+KGNN, +SAN, +TBE)	**72.25**	**31.05**	**53.25**	**18.29**

4 RELATED WORK

4.1 Session-based Recommendation

Session-based recommendation, which is an emerging topic in the recommendation system, has attracted interests of many researchers from both academia and industry. Hidasi et al. [4] use a gated recurrent unit model to model the user's behaviors in each single session. Li et al. [8] utilizes attention mechanisms to fusion the local interests and global users' preference. Liu et al. [11] further takes the long/short term memory into account when constructing a neural attention model.

4.2 Graph Neural Network for Session-based Recommendation

Graph Neural Networks(GNN), which can capture both graph structure and nodes' attributes in the graph, have shown its superiority in many applications. Song et al. [19] represents the interactions between users and items as a bipartite graph. Li et al. [9] proposed Gated-GSNN for sequential learning in graph structure. Wu et al. [23], which constructs item-item graph in each session, is a pioneering work that first use GNN to capture the nonlinear relationship between items in a session. It generates session embedding using attentive networks and predict the next item. Xu et al. [25] uses transformer to learn inter-item dependencies and generate session embedding for prediction.

5 CONCLUSION

In this paper, we present a novel architecture that incorporates knowledge graph and time interval information into session-based recommendation. The proposed method leverages knowledge-enhanced graph neural networks to obtain informative item embedding and uses temporal transformer to learn session representation effectively. Comprehensive experiments show that the proposed approach can consistently outperform state-of-art methods.

REFERENCES

[1] Ernie Chang, Jeriah Caplinger, Alex Marin, Xiaoyu Shen, and Vera Demberg. 2020. Dart: A lightweight quality-suggestive data-to-text annotation tool. In *Proceedings of the 28th International Conference on Computational Linguistics: System Demonstrations*. 12–17.

[2] Ernie Chang, Xiaoyu Shen, Dawei Zhu, Vera Demberg, and Hui Su. 2021. Neural Data-to-Text Generation with LM-based Text Augmentation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. 758–768.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

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

[5] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *ICDM*.

[6] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker. 2019. Time2Vec: Learning a Vector Representation of Time. *arXiv:1907.05321* (2019).

[7] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv:1609.02907* (2016).

[8] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *CIKM*.

[9] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2015. Gated graph sequence neural networks. *arXiv:1511.05493* (2015).

[10] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*.

[11] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *SIGKDD*.

[12] Siyuan Qiu, Binxia Xu, Jie Zhang, Yafang Wang, Xiaoyu Shen, Gerard De Melo, Chong Long, and Xiaolong Li. 2020. Easyaug: An automatic textual data augmentation platform for classification tasks. In *Companion Proceedings of the Web Conference 2020*. 249–252.

[13] Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2019. RepeatNet: A repeat aware neural recommendation machine for session-based recommendation. In *AAAI*.

[14] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*.

[15] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *WWW*.

[16] Xiaoyu Shen, Youssef Oualil, Clayton Greenberg, Mittul Singh, and Dietrich Klakow. 2017. Estimation of Gap Between Current Language Models and Human Performance. *Proc. Interspeech 2017* (2017), 553–557.

[17] Xiaoyu Shen, Hui Su, Wenjie Li, and Dietrich Klakow. 2018. Nexus network: Connecting the preceding and the following in dialogue generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4316–4327.

[18] Elena Smirnova and Flavian Vasile. 2017. Contextual sequence modeling for recommendation with recurrent neural networks. In *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems*.

[19] Weiping Song, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. 2019. Session-based social recommendation via dynamic graph attention networks. In *WSDM*.

[20] Hui Su, Xiaoyu Shen, Zhou Xiao, Zheng Zhang, Ernie Chang, Cheng Zhang, Cheng Niu, and Jie Zhou. 2020. Moviechats: Chat like humans in a closed domain. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 6605–6619.

[21] 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.

[22] 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 *NIPS*.

[23] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *AAAI*.

[24] Chaojun Xiao, Ruobing Xie, Yuan Yao, Zhiyuan Liu, Maosong Sun, Xu Zhang, and Leyu Lin. 2021. UPRec: User-Aware Pre-training for Recommender Systems. *arXiv preprint arXiv:2102.10989* (2021).

[25] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph contextualized self-attention network for session-based recommendation. In *IJCAI*.

[26] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2019. Self-attention with Functional Time Representation Learning. In *NIPS*.

[27] Fajie Yuan, Xiangnan He, Haochuan Jiang, Guibing Guo, Jian Xiong, Zhezhao Xu, and Yilin Xiong. 2020. Future data helps training: Modeling future contexts for session-based recommendation. In *Proceedings of The Web Conference 2020*. 303–313.

[28] Yang Zhao, Xiaoyu Shen, Wei Bi, and Akiko Aizawa. 2019. Unsupervised rewriter for multi-sentence compression. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2235–2240.