RATING RATE PREDICTION USING BEHAVIOUR SEQUENCE TRANSFORMER FOR MOVIE RATING IN OTT (OVER THE TOP)

[1]Gandikota Kethana, [2]Machavaram Manjudevika
[1,2]UG Student, [1,2]Department of Computer Science & Engineering, Geethanjali Institute of Science and Technology, Gangavaram, Andhra Pradesh, India

ABSTRACT
In recent years, the proliferation of online platforms and services has led to an abundance of user-generated ratings and reviews across various domains such as e-commerce, entertainment, and hospitality. Predicting user ratings accurately is crucial for personalized recommendation systems and business intelligence applications. This paper proposes a novel approach for rating rate prediction leveraging Behaviour Sequence Transformer (BST). The BST model effectively captures temporal dependencies in user behaviour sequences, allowing for more accurate predictions compared to traditional methods. We demonstrate the effectiveness of the proposed approach on real-world datasets, showcasing improvements in prediction accuracy and robustness.

INTRODUCTION
In an era dominated by Over-the-Top (OTT) platforms, predicting movie ratings has become paramount for both viewers and content creators. Introducing Behaviour Sequence Transformers (BST) – a cutting-edge methodology poised to revolutionize rating predictions in the OTT realm.
BST leverages the power of deep learning to analyze user behaviour sequences, decoding patterns and preferences with unparalleled precision. By scrutinizing viewing histories, engagement levels, and contextual cues, BST crafts a bespoke model for each user, ensuring tailored and accurate predictions.
Join us on an exploration of the future of rating predictions, where BST transcends traditional methodologies, offering a personalized and insightful glimpse into the world of OTT ratings.

DATAPRE-PROCESSING
Data preprocessing in a behaviour sequence transformer involves preparing the input data to be fed into the model. This typically includes steps like tokenizing the text data, encoding categorical variables, normalizing numerical features, handling missing values, and possibly augmenting the data for better model performance. The specific preprocessing steps depend on the nature of the data and the requirements of the transformer model being used Data preprocessing in behaviour sequence transformers typically involves several steps to prepare the input data for the model. Here's a general outline of the process:

a. Data Collection
b. Sequence Encoding
c. Sequence Padding
d. Feature Engineering
e. Normalization
f. Train-Test Split
g. Data Loading

ENCODE INPUT FEATURES
The encode_input_features method works as follows:

- Each categorical user feature is encoded using layers.Embedding, with embedding dimension equals to the square root of the vocabulary size of the feature. The embeddings of these features are concatenated to form a single input tensor.

- Each movie in the movie sequence and the target movie is encoded layers.Embedding, where the dimension size is the square root of the number of movies.
- A multi-hot genres vector for each movie is concatenated with its embedding vector, and processed using a non-linear layers.Dense to output a vector of the same movie embedding dimensions.
- A positional embedding is added to each movie embedding in the sequence, and then multiplied by its rating from the ratings sequence.
- The target movie embedding is concatenated to the sequence movie embeddings, producing a tensor with the shape of [batch size, sequence length, embedding size], as expected by the attention layer for the transformer architecture.
  The method returns a tuple of two elements:
  encoded_transformer_features and encoded_other_features.

The project focusing on rating rate prediction for movies in Over-The-Top (OTT) platforms using a Behaviour Sequence Transformer (BST) would likely involve collecting user behaviour data such as viewing history, ratings given to movies, and interactions with the platform. The BST would then be trained on this data to learn patterns in user behaviour sequences, enabling it to predict ratings for movies that users haven't yet rated.

The project might also involve additional steps such as feature engineering to extract relevant information from the data, hyperparameter tuning to optimize the BST model's performance, and evaluation using metrics like Mean Absolute Error or Root Mean Squared Error to measure the accuracy of the rating predictions.

Overall, the goal would be to create a predictive model that can accurately forecast user ratings for movies on OTT platforms, ultimately enhancing the recommendation systems and user experience.

OBJECTIVE

Ambiguity to upload a movie on an OTT platform to know about the rating rate giving to the movie by the user which will helps to the OTT platforms to know whether they will sustain on the OTT market.

LITERATURESURVEY

Self-Attentive Sequential Recommendation.

Sequential dynamics are a key feature of many modern recommender systems, which seek to capture the `context' of users' activities on the basis of actions they have performed recently. To capture such patterns, two approaches have proliferated: Markov Chains (MCs) and Recurrent Neural Networks (RNNs). Markov Chains assume that a user's next action can be predicted on the basis of just their last (or last few) actions, while RNNs in principle allow for longer-term semantics to be uncovered.

Generally speaking, MC-based methods perform best in extremely sparse datasets, where model parsimony is critical, while RNNs perform better in denser datasets where higher model complexity is affordable. The goal of our work is to balance these two goals, by proposing a self-attention based sequential model (SASRec) that allows us to capture long-term semantics (like an RNN), but, using an attention mechanism, makes its predictions based on relatively few actions (like an MC). At each time step, SASRec seeks to identify which items are `relevant' from a user's action history, and use them to predict the next item. Extensive empirical studies show that our method outperforms various state-of-the-art sequential models (including MC/CNN/RNN-based approaches) on both sparse and dense datasets.

Moreover, the model is an order of magnitude more efficient than comparable CNN/RNN-based models. Visualizations on attention weights also show how our model adaptively handles datasets with various density, and uncovers meaningful patterns in activity sequences.

Perceive Your Users in Depth: Learning Universal User Representations from Multiple E-commerce Tasks

Tasks such as search and recommendation have become increasingly important for E-commerce to deal with the information over- load problem. To meet the diverse needs of different users, personalization plays an important role. In many large portals such as Taobao and Amazon, there are a bunch of different types of search and recommendation tasks operating simultaneously for personalization.

However, most of current techniques address each task separately. This is suboptimal as no information about users shared across different tasks. In this work, we propose to learn universal user representations across multiple tasks for more effective personalization. In particular, user behaviour sequences (e.g., click, bookmark or purchase of products) are modelled by LSTM and attention mechanism by integrating all the corresponding content, behaviour and temporal information. User representations are shared and learned in an end-to-end setting across multiple tasks.

Bene ting from better information utilization of multiple tasks, the user representations are more effective to their interests and are more general to be transferred to new tasks. We refer this work as Deep User Perception Network (DUPN) and conduct an extensive set of online and online experiments. Across all tested different tasks, our DUPN consistently achieves better results by giving more effective user representations. Moreover, we deploy DUPN in large scale operational tasks in Taobao.

BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer

Modelling users' dynamic and evolving preferences from their historical behaviours is challenging and crucial for recommendation systems. Previous methods employ sequential neural networks (e.g., Recurrent Neural Network) to encode users' historical interactions from left to right into hidden representations for making recommendations. Although these methods achieve satisfactory results, they often assume a rigidly ordered sequence which is not always practical. We argue that such left-to-right unidirectional architectures restrict the power of the historical sequence representations.

For this purpose, we introduce a Bidirectional Encoder Representations from Transformers for sequential Recommendation (BERT4Rec). However, jointly conditioning on both left and right context in deep bidirectional model would make the training become trivial since each item can indirectly "see the target item". To address this problem, we train the bidirectional model using the Cloze task, predicting the masked items in the sequence by jointly conditioning on their left and right context.

Comparing with predicting the next item at each position in a sequence, the Cloze task can produce more samples to train a more powerful bidirectional model. Extensive experiments on four benchmark datasets show that our model outperforms various state-of-the-art sequential models consistently.

Existing System

MLP networks are used for supervised learning format. A typical learning algorithm for MLP networks is also called back propagation's algorithm.

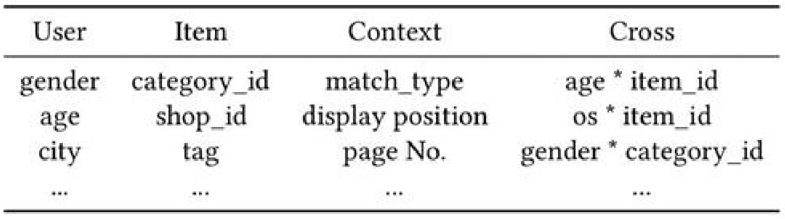
A multilayer perceptron (MLP) is a feed forward artificial neural network that generates a set of outputs from a set of inputs. An MLP is characterized by several layers of input nodes connected as a directed graph between the input nodes connected as a directed graph between the input and output layers. MLP uses backpropagation for training the network. MLP is a deep learning method

Proposed System

The overview architecture of the proposed BST. BST takes as input the user's behaviour sequence, including the target item, and "Other Features". It firstly embeds these input features as low-dimensional vectors. To better capture the relations among the items in the behaviour sequence, the transformer layer is used to learn deeper representation for each item in the sequence. Then by concatenating the embeddings of Other Features and the output of the transformer layer, the three-layer MLPs are used to learn the interactions of the hidden features, and sigmoid function is used to generate the final output.

EMBEDDING LAYER

The first component is the embedding layer, which embeds allinput features into a fixed-size low-dimensional vectors. In ourscenarios, there are various features, like the user profile features,item features, context features, and the combination of differentfeatures, i.e., the cross features1. Since this work is focused onmodeling the behaviour sequence with transformer, we denote allthese features as "Other Features" for simplicity, and give someexamples in Table 1. As shown in Figure 1, we concatenate "Otherfeatures" in left part and embed them into low-dimensional vectors.For these features, we create an embedding matrix $W_o \in R^{|D| \times d_o}$, where d_o is the dimension size.Besides, we also obtain the embeddings for each item in thebehaviour sequence, including the target item. As shown in Figure 1,we use two types of features to represent an item, "Sequence Item1Though the combination of features can be automatically learned by neural networks, we still incorporate the some hand-crafted cross features, which have beendemonstrated useful in our scenarios before the deep learning era.



User	Item	Context	Cross
gender	category_id	match_type	age * item_id
age	shop_id	display position	os * item_id
city	tag	page No.	gender * category_id
...	...	...	...

Table 1: The "Other Features" shown in left side of Figure.We use much more features in practice, and show a numberof effective ones for simplicity.

Features"(in red) and "Positional Features" (in dark blue), where"Sequence Item Features" include item_id and category_id. Notethat an item tends to have hundreds of features, while it is too expensive to chooseall to represent the item in a behaviour sequence.As introduced in our previous work , the item_id and category_id are good enough for the performance, we choose these twoas sparse features to represent each item in embedding the user'sbehaviour sequences. The "Positional Features" corresponds the following "positional embedding". Then for each item, we concatenateSequence Item Features and Positional Features, and create an embedding matrix $WV \in R|V|^{\times d}{}_{V}$, where d_V is the dimension size of
the embedding, and |V | is the number of items. We use $e_i \in RD^{v}$to represent the embedding for the i-th item in a given behavioursequence

TRANFORMER LAYER

In this part, we introduce the Transformer layer, which learns a deeper representation for each item by capturing the relations with other items in the behaviour sequences. Self-attention layer. The scaled dot-product attention is defined as follows:

To avoid overfitting and learn meaningful features hierarchically, we use dropout and LeakyReLU both in self-attention and FFN. Then the overall output of the self-attention and FFN layers are as follows:

$S' = \mathrm{LayerNorm}(S + \mathrm{Dropout}(MH(S))$

$F = \mathrm{LayerNorm} \quad S' + \mathrm{Dropout}(\mathrm{LeakyReLU}(S'W^{(1)} + b^{(1)})W^{(2)} + b^{(2)})$

Where $W^{(1)}$, $b^{(1)}$, $W^{(2)}$,$b^{(2)}$are the learnable parameters, and LayerNorm
is the standard normalization layer.
In practice, we observe in our experiments $b = 1$ obtains better
performance comparing to $b = 2, 3$ (see Table 4). For the sake of
efficiency, we did not try larger b and leave this for future work

System Architecture

The proposed Behaviour Sequence Architecture Theory (BSAT) is a psychological framework proposed by psychologist Mihaly Csikszentmihalyi. It suggests that optimal experiences, also known as "flow," occur when there is a balance between an individual's skills and the challenges they face. According to BSAT, achieving flow involves engaging in activities that are neither too easy nor too difficult for one's skill level, leading to a state of deep concentration and enjoyment. Csikszentmihalyi identified several components of flow, including clear goals, immediate feedback, and a merging of action and awareness.

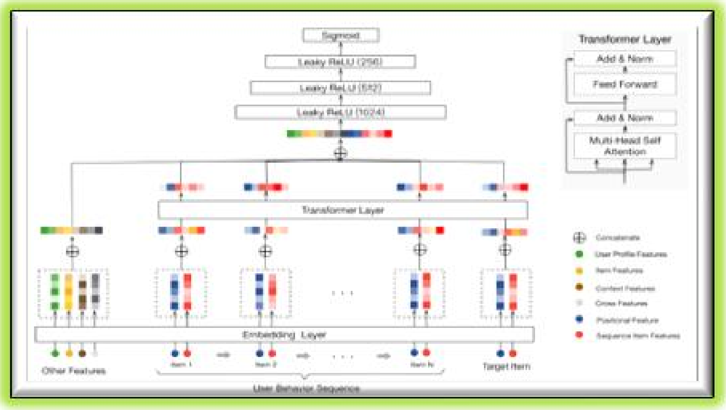


Figure.1. behaviour sequence transformer architecture

The above-mentioned fig 1shows the work flow of the system.The workflow of the system is discussed as follows:

They have used encoders of transformer architecture which uses self-attention to combine signals from users' past interactions. Self-attention is a pretty effective mechanism to capture any recent change in users' interests and also preserving long-term context at the same time.

Input Embedding

This layer reshapes various features from interaction, user, and items data and add them together to create a final input vector which would be input to the transformer's encoder layers.

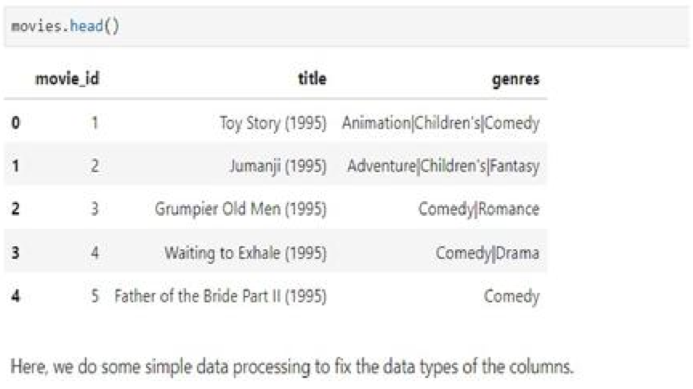
There are two types of features to represent an item, "Sequence Item Features"(in red) and "Positional Features" (in dark blue), where "Sequence Item Features" include item_id and category_id.

There are various features, like the user profile features, item features, context features, and the combination of different features, i.e., the cross features. Since this work is focused on modeling the behaviour sequence with transformer, they denote all these features as "Other Features".

Though the combination of features can be automatically learned by neural networks, they still incorporate some hand-crafted cross features, which have been demonstrated useful in our scenarios before the deep learning era.

DATASET

We use the 1M version of the Movielens dataset. The dataset includes around 1 million ratings from 6000 users on 4000 movies, along with some user features, movie genres. In addition, the timestamp of each user-movie rating is provided, which allows creating sequences of movie ratings for each user, as expected by the BST model.



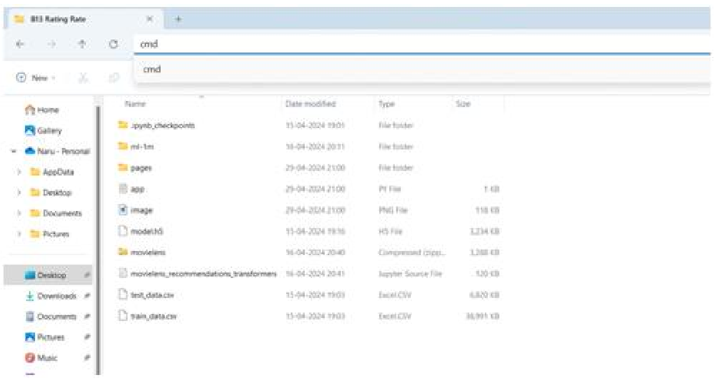
```
movies.head()
```

	movie_id	title	genres
0	1	Toy Story (1995)	Animation|Children's|Comedy
1	2	Jumanji (1995)	Adventure|Children's|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama
4	5	Father of the Bride Part II (1995)	Comedy

Here, we do some simple data processing to fix the data types of the columns.

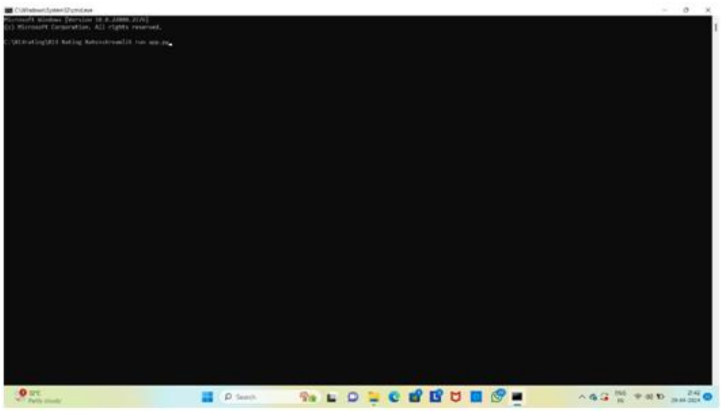
4.2 Fig: dataset that collected from movie lens

Results



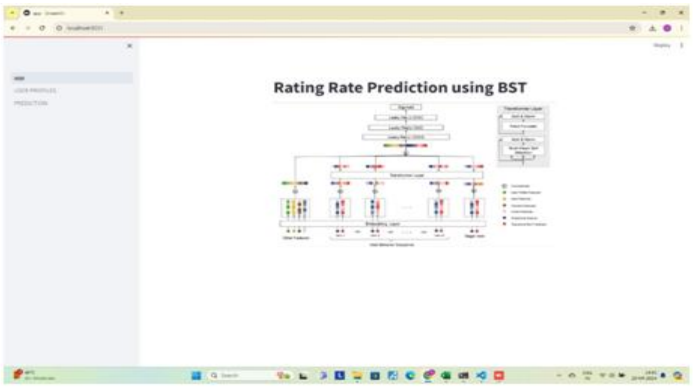
Name	Date modified	Type	Size
.ipynb_checkpoints	15-04-2024 19:01	File folder	
ml-1m	16-04-2024 20:11	File folder	
pages	29-04-2024 21:00	File folder	
app	29-04-2024 21:00	PY File	1 KB
image	29-04-2024 21:00	PNG File	118 KB
model.h5	15-04-2024 19:16	H5 File	3,234 KB
movielens	16-04-2024 20:40	Compressed (zipp...	3,288 KB
movielens_recommendations_transformers	16-04-2024 20:41	Jupyter Source File	120 KB
test_data.csv	15-04-2024 19:03	Excel CSV	6,820 KB
train_data.csv	15-04-2024 19:03	Excel CSV	38,991 KB

Step1: Open command prompt from the file location



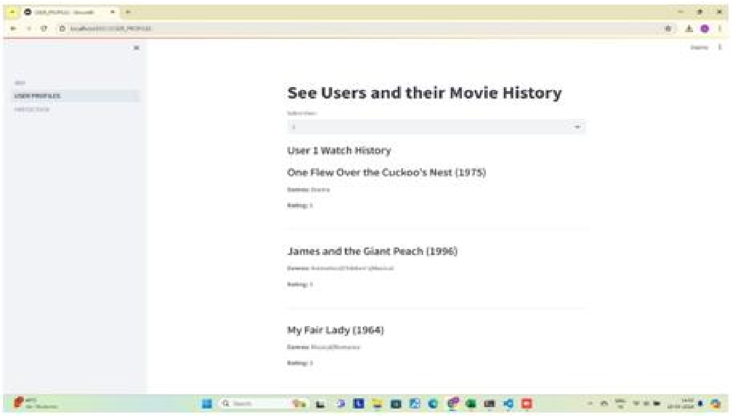
```
Microsoft Windows [Version 10.0.22000.2176]
(c) Microsoft Corporation. All rights reserved.

C:\[illegible]\B13 Rating Rate>streamlit run app.py
```

Step2: open the web application



Rating Rate Prediction using BST

Step3:   web application interface



See Users and their Movie History

User 1 Watch History

One Flew Over the Cuckoo's Nest (1975)

Genres: Drama

Rating: 5

James and the Giant Peach (1996)

Genres: Animation|Children's|Musical

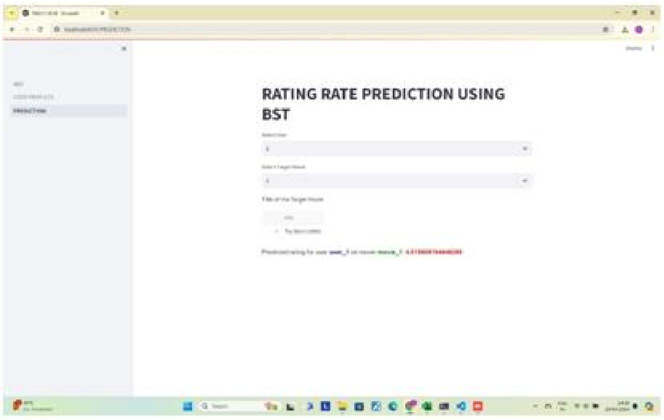
Rating: 3

My Fair Lady (1964)

Genres: Musical|Romance

Rating: 3

Step4: In above screen is showing user history


RATING RATE PREDICTION USING BST

Predicted rating for user user_[illegible] on movie movie_[illegible]: [illegible]

Step5: Showing the predicted output/rating

CONCLUSION

In this digital era,The BST model uses the Transformer layer in its architecture to capture the sequential signals underlying users' behaviour sequences for recommendation.

You can try training this model with different configurations, for example, by increasing the input sequence length and training the model for a larger number of epochs. In addition, you can try including other features like movie release year and customer zipcode, and including cross features.

FUTUREWORK

Use neural networks to learn complex user-item interactions by embedding users and movies into a latent space. Techniques like Matrix Factorization, Autoencoders, or Neural Collaborative Filtering can be employed for this purpose.

Incorporate temporal dynamics by considering the evolution of user preferences over time. Recurrent Neural Networks (RNNs) or Temporal Convolutional Networks (TCNs) can capture sequential patterns in user behaviour and adapt to changing preferences.

Utilize movie metadata such as genre, cast, director, and plot summary to build a content-based recommendation system. Natural Language Processing (NLP) techniques can be used to extract features from textual data, while traditional machine learning models or neural networks can make predictions based on these features.

Utilize movie metadata such as genre, cast, director, and plot summary to build a content-based recommendation system. Natural Language Processing (NLP) techniques can be used to extract features from textual data, while traditional machine learning models or neural networks can make predictions based on these

References

[1] Wen Chen, Pipei Huang, Jiaming Xu, Xin Guo, Cheng Guo, Fei Sun, Chao Li, Andreas Pfadler, Huan Zhao, and Binqiang Zhao. 2019. POG: Personalized Outfit Generation for Fashion Recommendation at Alibaba iFashion.

[2] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. , 7–10 pages.

[3] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In RecSys. 191–198.

[4] 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).

[5] Mihajlo Grbovic and Haibin Cheng. 2018. Real-time personalization using embeddings for search ranking at Airbnb. In KDD. 311–320.

[6] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In IJCAI. 1725–1731.

[7] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In ICDM. 197–206.

[8] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Pipei Huang, Huan Zhao, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-Interest Network with Dynamic Routing for Recommendation at Tmall.

[9] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xDeepFM: Combining explicit and implicit feature interactions for recommender systems. In KDD. 1754–1763.

[10] Yabo Ni, Dan Ou, Shichen Liu, Xiang Li, Wenwu Ou, Anxiang Zeng, and Luo Si. 2018. Perceive Your Users in Depth: Learning Universal User Representations from Multiple E-commerce Tasks. In KDD. 596–605.

[11] Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Wenwu Ou, and Dan Pei. 2019. Personalized Context-aware Reranking for E-commerce Recommender Systems. arXiv preprint arXiv:1904.06813 (2019).

[12] 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.

[13] 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. 5998–6008.

[14] Chenglong Wang, Feijun Jiang, and Hongxia Yang. 2017. A hybrid framework for text modeling with convolutional rnn. In KDD. 2061–2069.

[15] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In KDD. 839–848.

[16] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & Cross Network for Ad Click Predictions. In Proceedings of the ADKDD'17 (ADKDD'17).

[17] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In KDD. 1059–1068.

[18] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning Tree-based Deep Model for Recommender Systems. In KDD.
1079–1088.