PEAR: Personalized Re-ranking with Contextualized Transformer for Recommendation

Yi Li[1*], Jieming Zhu[2*], Weiwen Liu[2], Liangcai Su[1], Guohao Cai[2], Qi Zhang[2], Ruiming Tang[2]
Xi Xiao[1⋆], Xiuqiang He

[1]Shenzhen International Graduate School, Tsinghua University, China
[2]Huawei Noah's Ark Lab, China
{yi-li20, sulc21}@mails.tsinghua.edu.cn, xiaox@sz.tsinghua.edu.cn
{jamie.zhu, liuweiwen8, caiguohao1, zhangqi193, tangruiming, hexiuqiang1}@huawei.com

ABSTRACT

The goal of recommender systems is to provide ordered item lists to users that best match their interests. As a critical task in the recommendation pipeline, re-ranking has received increasing attention in recent years. In contrast to conventional ranking models that score each item individually, re-ranking aims to explicitly model the mutual influences among items to further refine the ordering of items given an initial ranking list. In this paper, we present a personalized re-ranking model (dubbed PEAR) based on contextualized transformer. PEAR makes several major improvements over the existing methods. Specifically, PEAR not only captures feature-level and item-level interactions, but also models item contexts from both the initial ranking list and the historical clicked item list. In addition to item-level ranking score prediction, we also augment the training of PEAR with a list-level classification task to assess users' satisfaction on the whole ranking list. Experimental results on both public and production datasets have shown the superior effectiveness of PEAR compared to the previous re-ranking models.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Recommender systems, personalized re-ranking, transformer

ACM Reference Format:
Yi Li, Jieming Zhu, Weiwen Liu, Liangcai Su, Guohao Cai, Qi Zhang, Ruiming Tang, Xi Xiao, Xiuqiang He. 2022. PEAR: Personalized Re-ranking with Contextualized Transformer for Recommendation. In *Companion Proceedings of the Web Conference 2022 (WWW '22 Companion), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3487553.3524208

[*] Both authors contributed equally to this work.
[⋆] Corresponding author.

WWW '22 Companion, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9130-6/22/04…$15.00
https://doi.org/10.1145/3487553.3524208

1 INTRODUCTION

Recommender systems are playing an increasingly important role in our daily lives, serving a wide variety of Web applications, such as e-commence, social networks, news feeds, etc. Due to the stringent latency requirements, industrial recommender systems typically follow a matching→ranking→re-ranking pipeline to gradually reduce the number of candidate items from millions to thousands, to hundreds, and finally to tens. As the final goal of recommender systems is to provide an ordered item list to users, re-ranking becomes a critical task and has received increasing attention in recent years [1, 16, 23, 28].

Most of the ranking models only consider the features of each user-item pair individually and learn a scoring function via a pointwise [6], pairwise [3] or listwise [22] loss. However, these models fail to explicitly model the mutual influences between items in the feature space. In contrast, re-ranking aims to model the scoring function by jointly encoding a list of items into contextualized representations, since the score of an item often depends on the other items placed in the same list [16]. This results in a more complex model architecture, which is usually used to refine the ordering of tens (e.g., top 30) of items in a given initial ranking list.

Several pioneer research efforts [1, 16, 23, 28] have been made to achieve this goal. Concretely, GlobalRank [28] and DLCM [1] employ recurrent neural networks (RNNs) to encode the initial ranking list sequentially or bidirectionally. But RNNs have limited capacity in capturing pairwise item relationships. Later studies, such as PRM [16] and SetRank [15], apply the self-attention mechanism in transformers to modeling inter-dependencies among items. Transformer architecture is more favorable due to its effectiveness in modeling interactions of any two items and its efficiency in parallelization. Although these models have focused on re-ranking modeling, they fail to take full advantage of the unique characteristic in recommender systems, that is personalization. Especially, many existing studies evaluate the re-ranking effectiveness using Web search datasets [1, 15, 16, 21]. We argue that in recommender systems, the ranking of an item depends on not only the context of initial ranking list but also the context of historical clicked item list of a user. For example, if there are baby products and trendy clothing in the historical clicked list of a mother whose child is still infancy, then baby products may be ranked before the trendy clothing. Note that although user behavior sequence has been widely studied in other recommendation tasks, such as sequential recommendation for matching [10, 18] and click-through rate (CTR) prediction for ranking [4, 25], it is still unexplored for personalized re-ranking.

To address the issue, in this work, we propose a personalized re-ranking model (dubbed PEAR) based on a novel contextualized transformer architecture, which comprises three main parts: feature-level interaction, item-level interaction, and multi-task training. More specifically, our PEAR model has made the following major improvements over the existing re-ranking models.

- We model both feature-level and item-level interactions to combine the advantages of both ranking and re-ranking models. The feature interaction module can be substituted to any existing ranking network that allows to capture feature interactions, such as MLP [8], DCN [20], AFN [7].
- We not only leverage the contextual information within the initial ranking list as existing studies have done, but also capture the item inter-dependencies across the historical clicked item list via merged sequence-to-sequence cross-attention, which expands the context scope for improved personalization in re-ranking.
- We train PEAR through an item-level ranking score prediction task as well as a list-level classification task to assess users' satisfaction on the whole ranking list, which follows the multi-task training paradigm.

To evaluate the effectiveness of PEAR, we have conducted comprehensive experiments on a public benchmark dataset for micro-video recommendation [5] and a large-scale production dataset for news recommendation at Huawei. The results show the superiority of PEAR over the existing state-of-art methods.

2 RELATED WORK

Transformer-based ranking. The great success of transformers in natural language processing (NLP) [11, 14, 19] draw much research attention from the recommendation domain. In recommendation tasks, user's historical behaviors with sequential structure is an important aspect for modeling. Therefore, many studies [4, 24–26] use the self-attention mechanism in transformers to model relations among items. Zhou et al. [26] introduces the target attention mechanism to adaptively assign importance weights to item embeddings in historical behaviors according to the target item for ranking. To better capture item relationships and sequential patterns underlying the historical behavior sequences, Chen et al. [4] applies the transformer encoder to learn contextualized item representations of historical behaviors. Instead, our work leverages transformers for re-ranking.

Re-ranking. Different from the ranking stage where each item score is computed independently, re-ranking takes into account the mutual influences of items generated from a ranking model to refine their ordering. Several pioneer studies on re-ranking have been performed. [28] and [1] utilize RNNs to encode item-level sequential patterns into feature space. However, RNNs are limited in modeling pairwise relations directly, especially for long sequences. In contrast, PRM [16] introduces the self-attention mechanism to model inter-item dependencies in initial ranking list for re-ranking. In addition, [13] proposes a graph neural network based model to capture item relationships. Qiu et al. [17] investigate the use of data augmentation to mitigate the imbalance issue in re-ranking. Different from these models, our work leverages contextualized transformers to incorporate historical behaviors for re-ranking.

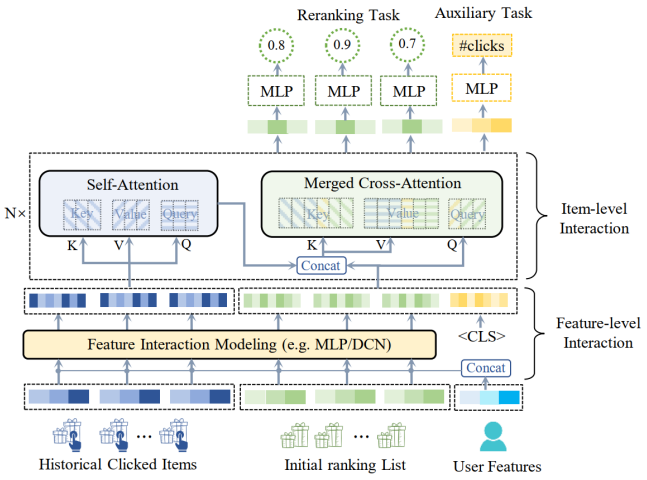

Figure 1: The model architecture of PEAR.

3 METHODOLOGY

In this section, we describe the PEAR model architecture as illustrated in Figure 1. There are three main parts in the framework: feature-level interaction, item-level interaction and multi-task training. In what follows, we describe each part in details.

3.1 Feature-level Interaction

Learning interactions of raw features is important as indicated in existing studies for ranking [9, 20]. To fuse features (e.g., gender, city, category) of each user-item pair, we propose a feature interaction module to capture complex feature interactions and thus obtain contextualized item representations. In recent years, many effective feature interaction networks have been proposed, such as DeepFM [9], DCN [20], and AFN [7], to capture useful feature interactions. While it is generally applicable to employ any of these networks as our feature interaction module, in this work, we use a two-layer MLP for simplicity.

Typically, both user features and item features comprise multi-field categorical values (e.g., the city field contains Beijing/Shanghai/$\cdots$). It is common to apply embedding layers to map these highly sparse categorical features into dense feature embeddings (as detailed in [27]). After that, we could obtain a flattened feature vector for each user ($f_u \in \mathbb{R}^{d_u}$) and item ($f_i \in \mathbb{R}^{d_i}$), where d_u and d_i denote flattened dimensions. To get fused user-item representations, we first concatenate the user feature vector and every item feature vector in the historical item list $B = [b_1, b_2, ..., b_n]$ and the initial ranking list $S = [s_1, s_2, ..., s_m]$, which produces the representation matrix $X \in \mathbb{R}^{(d_u+d_i)\times(n+m)}$. Then, we feed X into a two-layer MLP:

$$Z = W_2\,\sigma(W_1 X + b_1) + b_2\,, \tag{1}$$

where W_1, W_2, b_1, b_2 are learned kernel and bias weights. σ denotes the activation function (e.g., ReLU). $Z \in \mathbb{R}^{d\times(n+m)}$ outputs the fused representations in d-dimension. It is worth noting that our feature interaction module (i.e., 2-layer MLP) works similarly to the feedforward network (FFN) in transformers [19]. In addition, to facilitate the subsequent list-level task, we add a special classification token (i.e., CLS) with learnable parameters in $\mathbb{R}^d$ at the end

of the initial ranking list. It is inspired by Bert [11], and is useful in aggregating the list-level representation of the entire initial ranking list. Finally, we obtain $Z_B \in \mathbb{R}^{d\times n}$ and $Z_S \in \mathbb{R}^{d\times(m+1)}$ as the output representations of the historical item list and the initial ranking list (with CLS token), respectively.

3.2 Item-level Interaction

Several pioneer research efforts [1, 2, 16, 28] have been made to model the mutual influences between item pairs. Although they introduce different network structures (e.g., RNNs, self-attention) for item relation modeling, all of them only consider the context of the initial ranking list, largely neglecting the potential influences by the historical clicked item list. However, historical user behaviors contain rich and fine-grained user interests that are equally important for contextualized item modeling in re-ranking.

In this work, we employ the popular encoder-decoder structure [19] that has been widely applied in NLP to capture the item-level interactions not only within the initial ranking list and within the historical item list, but also across the two lists. As shown in Figure 1, the item-level interaction module consists of a stack of N blocks, each comprising a self-attention layer and a merged cross-attention layer. For efficiency, we set $N = 1$ in our experiments.

First, a self-attention layer is employed to model fine-grained user interests in the historical behaviors, which could provide more informative contexts for learning item interactions across lists. Specifically, we use the self-attention in [19] as follows:

$$H_B = Softmax\left(\frac{(W_Q Z_B)^T (W_K Z_B)}{\sqrt{d_h}}\right)(W_V Z_B)^T, \tag{2}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d_h\times d}$ are learnable query, key, and value parameters. $H_B \in \mathbb{R}^{n\times d_h}$ represents the output item representations in historical item list.

Then, a merged cross-attention layer is applied to model the item interactions within the initial ranking list and across the two lists simultaneously. For this purpose, a straightforward way is to apply a self-attention sub-layer and a cross-attention sub-layer accordingly, as the transformer decoder does in [19]. But we further merge the two sub-layers together for better computation efficiency in GPUs. More specifically, given H_B and Z_S as inputs, the merged cross-attention is formulated as follows:

$$H_S = Softmax\left(\frac{(W_q Z_S)^T [W_{k1} H_B^T, W_{k2} Z_S]}{\sqrt{d_h}}\right)[W_{v1} H_B^T, W_{v2} Z_S]^T \tag{3}$$

where $W_q, W_{k2}, W_{v2} \in \mathbb{R}^{d_h\times d}$ and $W_{k1}, W_{v1} \in \mathbb{R}^{d_h\times d_h}$ are all learnable parameters. $[\cdot]$ denotes the concatenation operation. In this way, we use a single merged cross-attention layer to simultaneously capture item interactions within Z_S and across Z_S and H_B.

Our PEAR model comprises a feature-level interaction module and an item-level interaction module, corresponding to FFN and multi-head self-attention layers respectively in a vanilla transformer [19]. Thus, we view PEAR as a contextualized transformer.

3.3 Multi-task Training

The output H_S goes through a one-layer MLP followed by sigmoid activations for click prediction. It is worth noting that softmax loss does not work well since multiple positives may exist in the initial ranking list, making the sum of ground-truth probabilities larger than 1. Instead, we model the task as a multi-label learning problem using binary cross-entropy loss, which is defined as:

$$\mathcal{L}_m = \sum \Big(y_t log(\hat{y}_t) + (1 - y_t) log(1 - \hat{y}_t)\Big) \tag{4}$$

where $y_t \in \{0, 1\}$ denotes whether the user has clicked the target item. $\hat{y}_t$ is the predicted click probability of the target item.

Considering that the above loss only considers each single item in isolation, we introduce an auxiliary task, which is a list-level task to predict the number of clicks in each initial list. For simplicity, in this work, we also formulate it as a binary classification problem with the following loss:

$$\mathcal{L}_{aux} = \sum \Big(y_{aux} log(\hat{y}_{aux}) + (1 - y_{aux}) log(1 - \hat{y}_{aux})\Big) \tag{5}$$

where y_{aux} represents whether the initial ranking list contains positive items. $\hat{y}_{aux}$ is the predicted probability from the CLS token. At last, we combine the two loss functions as multi-task learning:

$$\mathcal{L} = \mathcal{L}_m + \alpha \mathcal{L}_{aux} \tag{6}$$

where α is a hyper-parameter to adjust the effect of $\mathcal{L}_{aux}$. Typically, we set $\alpha = 1$.

4 EXPERIMENTS

4.1 Experimental Settings

4.1.1 Dataset Description. Existing works conduct experiments on Web search datasets [1, 15, 16, 21]. However, these datasets do not contain user historical behaviors required by our work. Thus, we employ another two real-world datasets in our experiments:

- **MicroVideo-1.7M**: This is a million-scale dataset collected from a popular micro-video recommender system in China, which has 12,737,619 interactions that 10,986 users have made on 1,704,880 micro-videos [5]. We follow the existing train-test data splits.
- **News-Dataset**: The private dataset contains click logs of three hours, randomly sampled from Huawei's production news recommender systems. It is composed of 17,203,522 interactions between 1.8M users and 37K news.

4.1.2 Evaluation Metrics. We perform re-ranking for 30 items and use the following common metrics for evaluating re-ranking performance [1, 26], including nDCG@K and gAUC@K, where K=20, 30.

4.1.3 Initial ranker and baselines. To generate an initial ranking list for the re-ranking model, we select DCN, a learning to rank method that achieves great success in industry. And we compare our proposed PEAR against several baseline re-ranking models including DLCM [1], PRM [16] and SetRank [15].

4.1.4 Implementation details. We use Tensorflow for model implementation and apply the Adam [12] optimizer. The maximal length of historical behaviors is set to 128. The MLP used for feature-level interaction has a size of [500, 500]. We set the dropout rate to 0.1, batch size to 200 for MicroVideo-1.7M and 400 for News-Dataset. We tune the learning rate from {1e-3, 1e-4, 5e-5, 1e-5}. The number of heads is set 1 for MicroVideo-1.7M and 2 for News-Dataset.

Table 1: Performance comparison of PEAR to other re-ranking methods.

Model	MicroVideo-1.7M				News-Dataset			
	gAUC@20	gAUC@30	nDCG@20	nDCG@30	gAUC@20	gAUC@30	nDCG@20	nDCG@30
DCN (base)	0.5342	0.5405	0.5349	0.6585	0.6014	0.6097	0.2557	0.2601
DLCM	0.5604	0.5665	0.5571	0.6711	0.6040	0.6115	0.2564	0.2608
SetRank	0.5483	0.5537	0.5445	0.6620	0.6017	0.6099	0.2555	0.2602
PRM	0.5629	0.5680	0.5579	0.6714	0.6059	0.6125	0.2572	0.2616
PEAR	**0.5681**	**0.5741**	**0.5640**	**0.6755**	**0.6172**	**0.6230**	**0.2599**	**0.2641**

Table 2: Effect of multi-task training on MicroVideo-1.7M

Model	gAUC@20	gAUC@30	nDCG@20	nDCG@30
w/o auxiliary task	0.5606	0.5672	0.5626	0.6747
w/ auxiliary task	**0.5681**	**0.5741**	**0.5640**	**0.6755**

The hidden dimensionality d_h is 500. We apply early stopping (patience=2) to avoid overfitting during training. We also tune the baseline models exhaustively to get reasonable results.

4.2 Performance Comparison

The overall performance of our baselines and PEAR are shown in Table 1. For each dataset, we report the result of one ranking model DCN, three re-ranking sota baselines and PEAR. The best performance are highlighted in bold.

Results show that PEAR consistently outperforms all baselines across the public dataset and the industry dataset in all cases. Specifically, it achieves improvements over the strongest baselines in terms of gAUC@30 by 1.07%, 1.71%, in MicroVideo-1.7M and Huawei-Dataset, respectively. As for nDCG, the highest nDCG@20 of baselines reported on MicroVideo-1.7M is 0.5579. While our PRM can reach 0.5640, which achieve 1.09% improvements. This validates the effectiveness of our architecture for modeling historical behaviors in the re-ranking scenario. We owe this to the modeling of the dependency between the initial list and the historical list. This kind of explicit modeling between sequences can dig out the information in the historical behaviors that directly guides the reordering of the initial list. It is worth noting that PRM that does not regard the historical behaviors as a sequence for modeling is the most competitive baseline. One possible reason is that PRM benefits a lot from its personalized vector generated from historical behavior through a pre-training model, and this information is not used in other baselines. Nonetheless, because of the way it handles historical behaviors, the improvements of PRM brought by it is limited. In addition, noted that SetRank performs worst in the baseline models, which may be due to the attention rank loss adopted by SetRank is not suitable for recommendation task.

4.3 Ablation Study

Effect of auxiliary task: To prove the efficiency of the auxiliary task, we compare the performance of our model without and with auxiliary task on MicroVideo-1.7M dataset. Results are shown in Table 2. According to the results, the nDCG@30 drops from 0.5741 to 0.5672 when removing the auxiliary task. This verifies the effect of the global supervision signal of the list.

Effect of length of historical item list: Figure 2 shows the performance comparison of PEAR with different history lengths on the MicroVideo-1.7M dataset. Due to the space limitation, the results of gAUC@30 and nDCG@30 are selected for reporting. Obviously, as the length of historical sequence increases, the performance of our model is gradually improving. When the length of the historical sequence changes from 64 to 128, the performance improvement is most significant. This justifies the effectiveness of the sequence signal of historical behaviors in the re-ranking stage.

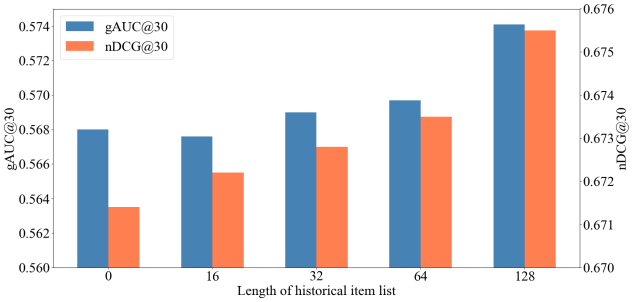

Figure 2: Effect of history length on MicroVideo-1.7M

Therefore, the above ablations show that each component in PEAR is effective in refining the order of the initial ranking list.

5 CONCLUSION

In this paper, we focus on personalized re-ranking with a contextualized transformer model, namely PEAR. In contrast to existing re-ranking models, our work makes three main improvements, including (i) modeling both feature-level and item-level interactions, (ii) capturing the contextual information within initial ranking list as well as across historical clicked item list, and (iii) applying multi-task learning to augment the training of PEAR with a list-level classification task. Extensive experiments have been conducted on both a public micro-video recommendation dataset and a production news recommendation dataset from Huawei to validate the effectiveness of our PEAR model. Future work might include improving model efficiency and better considering the cases for long-tail users (whose historical item list is too short).

6 ACKNOWLEDGMENTS

This work is partially supported by the National Natural Science Foundation of China (61972219), the Research and Development Program of Shenzhen (JCYJ20190813174403598, SGDX20190918101201696), the National Key Research and Development Program of China (2018YFB1800601), and the Overseas Research Cooperation Fund of Tsinghua Shenzhen International Graduate School (HW2021013).

REFERENCES

[1] Qingyao Ai, Keping Bi, Jiafeng Guo, and W Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 135–144.

[2] Irwan Bello, Sayali Kulkarni, Sagar Jain, Craig Boutilier, Ed Chi, Elad Eban, Xiyang Luo, Alan Mackey, and Ofer Meshi. 2018. Seq2slate: Re-ranking and slate optimization with rnns. *arXiv preprint arXiv:1810.02019* (2018).

[3] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*. 129–136.

[4] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[5] Xusong Chen, Dong Liu, Zheng-Jun Zha, Wengang Zhou, Zhiwei Xiong, and Yan Li. 2018. Temporal hierarchical attention at category-and item-level for micro-video click-through prediction. In *Proceedings of the 26th ACM international conference on Multimedia*. 1146–1153.

[6] 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. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[7] Weiyu Cheng, Yanyan Shen, and Linpeng Huang. 2020. Adaptive Factorization Network: Learning Adaptive-Order Feature Interactions. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*. 3609–3616.

[8] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.

[9] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[10] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *IEEE International Conference on Data Mining (ICDM)*. 197–206.

[11] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*. 4171–4186.

[12] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[13] Weiwen Liu, Qing Liu, Ruiming Tang, Junyang Chen, Xiuqiang He, and Pheng Ann Heng. 2020. Personalized Re-ranking with Item Relationships for E-commerce. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 925–934.

[14] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[15] Liang Pang, Jun Xu, Qingyao Ai, Yanyan Lan, Xueqi Cheng, and Jirong Wen. 2020. Setrank: Learning a permutation-invariant ranking model for information retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 499–508.

[16] Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Junfeng Ge, Wenwu Ou, et al. 2019. Personalized re-ranking for recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 3–11.

[17] Zi-Hao Qiu, Ying-Chun Jian, Qing-Guo Chen, and Lijun Zhang. 2021. Learning to Augment Imbalanced Data for Re-ranking Models. In *Proceeddings of the 30th ACM International Conference on Information and Knowledge Management (CIKM)*, Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong (Eds.). 1478–1487.

[18] 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 (CIKM)*. 1441–1450.

[19] 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.

[20] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. 1–7.

[21] Yunjia Xi, Weiwen Liu, Xinyi Dai, Ruiming Tang, Weinan Zhang, Qing Liu, Xiuqiang He, and Yong Yu. 2021. Context-aware Reranking with Utility Maximization for Recommendation. *CoRR* abs/2110.09059 (2021).

[22] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*. 1192–1199.

[23] Dawei Yin, Yuening Hu, Jiliang Tang, Tim Daly, Mianwei Zhou, Hua Ouyang, Jianhui Chen, Changsung Kang, Hongbo Deng, Chikashi Nobata, et al. 2016. Ranking relevance in yahoo search. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 323–332.

[24] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. 2018. Atrank: An attention-based user behavior modeling framework for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[25] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

[26] 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 *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1059–1068.

[27] Jieming Zhu, Jinyang Liu, Shuai Yang, Qi Zhang, and Xiuqiang He. 2021. Open Benchmarking for Click-Through Rate Prediction. In *The 30th ACM International Conference on Information and Knowledge Management (CIKM)*. 2759–2769.

[28] Tao Zhuang, Wenwu Ou, and Zhirong Wang. 2018. Globally optimized mutual influence aware ranking in e-commerce search. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 3725–3731.