

Personalized Re ranking for Recommendation

Changhua Pei^{1*}, Yi Zhang^{1*}, Yongfeng Zhang^{2*}

Fei Sun¹, Xiao Lin¹, Hanxiao Sun¹, Jian Wu¹, Peng Jiang³, Junfeng Ge¹, Wenwu Ou¹

¹ Alibaba Group ² Rutgers University ³ Kwai Inc.

¹ changhua.pch, zhanyuan.zy, ofey.sf, hc.lx, hansel.shx, joshuawu.wujian, beili.gjf, santong.oww}@alibaba-inc.com

² yongfeng.zhang@rutgers.edu ³ jiangpeng@kuaishou.com

ABSTRACT

Ranking is a core task in recommender systems, which aims at providing an ordered list of items to users. Typically, a ranking function is learned from the labeled dataset to optimize the global performance, which produces a ranking score for each individual item. However, it may be sub-optimal because the scoring function applies to each item individually and does not explicitly consider the mutual influence between items, as well as the differences of users' preferences or intents. Therefore, we propose a personalized re-ranking model for recommender systems. The proposed re-ranking model can be easily deployed as a follow-up modular after any ranking algorithm, by directly using the existing ranking feature vectors. It directly optimizes the whole recommendation list by employing a transformer structure to efficiently encode the information of all items in the list. Specifically, the Transformer applies a self-attention mechanism that directly models the global relationships between any pair of items in the whole list. We confirm that the performance can be further improved by introducing pre-trained embedding to learn personalized encoding functions for different users. Experimental results on both offline benchmarks and real-world online e-commerce systems demonstrate the significant improvements of the proposed re-ranking model.

CCS CONCEPTS

•Information systems → Recommender systems;

KEYWORDS

Learning to rank; Re-ranking; Recommendation

ACM Reference format:

Changhua Pei^{1*}, Yi Zhang^{1*}, Yongfeng Zhang² and Fei Sun¹, Xiao Lin¹, Hanxiao Sun¹, Jian Wu¹, Peng Jiang³, Junfeng Ge¹, Wenwu Ou¹. 2019. Personalized Re-ranking for Recommendation. In *Proceedings of Thirteenth ACM Conference on Recommender Systems, Copenhagen, Denmark, September 16–20, 2019 (RecSys '19)*, 9 pages. DOI: 10.1145/3298689.3347000

*Changhua Pei and Yi Zhang contribute equally. Yongfeng Zhang is the corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

RecSys '19, Copenhagen, Denmark

© 2019 ACM. 978-1-4503-6243-6/19/09...\$15.00

DOI: 10.1145/3298689.3347000

1 INTRODUCTION

Ranking is crucial in recommender systems. The quality of the ranked list given by a ranking algorithm has a great impact on users' satisfaction as well as the revenue of the recommender systems. A large amount of ranking algorithms [4, 5, 7, 15, 19, 27, 32] have been proposed to optimize the ranking performance. Typically ranking in recommender system only considers the user-item pair features, without considering the influences from other items in the list, especially by those items placed alongside [8, 35]. Though *pairwise* and *listwise* learning to rank methods try to solve the problem by taking the item-pair or item-list as input, they only focus on optimizing the loss function to make better use of the labels, e.g., click-through data. They didn't explicitly model the mutual influences between items in the feature space.

Some works [1, 34, 37] tend to model the mutual influences between items explicitly to refine the initial list given by the previous ranking algorithm, which is known as re-ranking. The main idea is to build the scoring function by encoding intra-item patterns into feature space. The state-of-the-art methods for encoding the feature vectors are RNN-based, such as GlobalRerank [37] and DLCM [1]. They feed the initial list into RNN-based structure sequentially and output the encoded vector at each time step. However, RNN-based approaches have limited ability to model the interactions between items in the list. The feature information of the previous encoded item degrades along with the encoding distance. Inspired by the Transformer architecture [20] used in machine translation, we propose to use the Transformer to model the mutual influences between items. The Transformer structure uses self-attention mechanism where any two items can interact with each other directly without degradation over the encoding distance. Meanwhile, the encoding procedure of Transformer is more efficient than RNN-based approach because of parallelization.

Besides the interactions between items, personalized encoding function of the interactions should also be considered for re-ranking in recommender system. Re-ranking for recommender system is user-specific, depending on the user's preferences and intents. For a user who is sensitive to price, the interaction between "price" feature should be more important in the re-ranking model. Typical global encoding function may be not optimal as it ignores the differences between feature distributions for each user. For instance, when users are focusing on price comparison, similar items with different prices tend to be more aggregated in the list. When the user has no obvious purchasing intention, items in the recommendation list tend to be more diverse. Therefore, we introduce a personalization module into the Transformer structure to represent user's preference and intent on item interactions. The interaction between

items in the list and user can be captured simultaneously in our personalized re-ranking model.

The main contributions of this paper are as follows:

- **Problem.** We propose a personalized re-ranking problem in recommender systems, which, to the best of our knowledge, is the first time to explicitly introduce the personalized information into re-ranking task in large-scale online system. The experimental results demonstrate the effectiveness of introducing users' representation into list representation for re-ranking.
- **Model.** We employ the Transformer equipped with personalized embedding to compute representations of initial input ranking list and output the re-ranking score. The self-attention mechanism enable us to model user-specific mutual influences between any two items in a more effective and efficient way compared with RNN-based approaches.
- **Data.** We release a large scale dataset (E-commerce Re-ranking dataset) used in this paper. This dataset is built from a real-world E-commerce recommender system. Records in the dataset contain a recommendation list for user with click-through labels and features for ranking.
- **Evaluation.** We conducted both offline and online experiments which show that our methods significantly outperform the state-of-the-art approaches. The online A/B tests show that our approach achieves higher click-through rate and more revenue for real-world system.

2 RELATED WORK

Our work aims to refine the initial ranking list given by the base ranker. Among these base rankers, learning to rank is one of the widely used methods. The learning to rank methods can be classified into three categories according to the loss function they used: *pointwise*[12, 21], *pairwise*[6, 18, 19], and *listwise*[5, 7, 14, 19, 27, 32, 33]. All these methods learn a global scoring function within which the weight of a certain feature is globally learned. However, the weights of the features should be able to be aware of the interactions not only between items but also between the user and items.

Closest to our work are [1–3, 37], which are all re-ranking methods. They use the whole initial list as input and model the complex dependencies between items in different ways. [1] uses unidirectional GRU[10] to encode the information of the whole list into the representation of each item. [37] uses LSTM[17] and [3] uses pointer network[29] not only to encode the whole list information, but also to generate the ranked list by a decoder. For those methods which use either GRU or LSTM to encode the dependencies of items, the capacity of the encoder is limited by the encoding distance. In our paper, we use transformer-like encoder, based on self-attention mechanism to model the interactions for any of two items in $O(1)$ distance. Besides, for those methods which use decoder to sequentially generate the ordered list, they are not suitable for online ranking system which requires strict latency criterion. As the sequential decoder uses the item selected at time $t-1$ as input to select the item at time t , it can not be parallelized and needs n times of inferences, where n is the length of the output list. [2] proposes a *groupwise* scoring function which can be parallelized

Table 1: Notation used in this paper.

| Notation. | Description. |
|---------------------------------|---------------------------------------------------------------------------------------------|
| X | The matrix of features. |
| PV | The matrix of personalized vectors. |
| P | The matrix of position embeddings. |
| \mathcal{R} | The set of total users' requests. |
| \mathcal{I}_r | The set of candidate items for each user's request $r \in \mathcal{R}$. |
| \mathcal{S}_r | The initial list of items generated by the ranking approaches for each user's request r . |
| \mathcal{H}_u | The sequence of items clicked by user u . |
| $\theta, \hat{\theta}, \theta'$ | The parameter matrices of ranking, re-ranking and pre-trained model respectively. |
| y_i | The label of click on item i . |
| $P(y_i \cdot)$ | The click probability of item i predicted by the model. |

when scoring the items, but its computation cost is high because it enumerates every possible combinations of items in the list.

3 RE-RANKING MODEL FORMULATION

In this section, we first give some preliminary knowledge about learning to rank and re-ranking methods for recommendation systems. Then we formulate the problem we aim to solve in this paper. The notations used in this paper are in Table 1.

Learning to rank (often labelled as **LTR**) method is widely used for ranking in real-work systems to generate an ordered list for information retrieval[18, 22] and recommendation[14]. The LTR method learns a global scoring function based on the feature vector of items. Having this global function, the LTR method outputs an ordered list by scoring each item in the candidate set. This global scoring function is usually learned by minimizing the following loss function \mathcal{L} :

$$\mathcal{L} = \sum_{r \in \mathcal{R}} \ell \{y_i, P(y_i|\mathbf{x}_i; \theta) | i \in \mathcal{I}_r\} \quad (1)$$

where \mathcal{R} is the set of all users' requests for recommendation. \mathcal{I}_r is the candidate set of items for request $r \in \mathcal{R}$. \mathbf{x}_i represents the feature space of item i . y_i is the label on item i , i.e., click or not. $P(y_i|\mathbf{x}_i; \theta)$ is the predicted click probability of item i given by the ranking model with parameters θ . ℓ is the loss computed with y_i and $P(y_i|\mathbf{x}_i; \theta)$.

However, \mathbf{x}_i is not enough to learn a good scoring function. We find that ranking for recommender system should consider the following extra information: (a) mutual influences between item-pairs[8, 35]; (b) interactions between the users and items. The mutual influences between item-pairs can be directly learned from the initial list $\mathcal{S}_r = [i_1, i_2, \dots, i_n]$ given by the existing LTR model for the request r . Works[1][37][2][3] propose approaches to make better use of mutual information of item-pairs. However, few works consider the interactions between the users and items. The extent of mutual influences of item-pairs varies from user to user. In this paper, we introduce a personalized matrix PV to learn user-specific encoding function which is able to model personalized mutual influences between item-pairs. The loss function of the model can