TMRM: Two-stage Multi-task Recommendation Model Boosted Feature Selection

Fan Zhu
Beijing Key Lab of Intelligent
Telecommunication Software and Multimedia
Beijing University of
Posts and Telecommunications
Beijing, China
zhf@bupt.edu.cn

Juan Yang*
Beijing Key Lab of Intelligent
Telecommunication Software and Multimedia
Beijing University of
Posts and Telecommunications
Beijing, China
yangjuan@bupt.edu.cn

Pengfei Wang
dept.computer
Beijing University of
Posts and Telecommunications
Beijing, China
wangpengfei@bupt.edu.cn

***Abstract*—For recommendation systems, full learning from features is critical to improving system accuracy. Most recommendation systems collect a large number of features to improve the recommendation accuracy, but they ignore the importance of extracting important feature combinations. Even if some systems manually measure important features based on experience, they cannot automatically select important feature combinations, that is why they can not be used to guide model training. In this paper, a Two-stage Multi-task Recommendation Model(TMRM) is proposed which aims to automatically select important feature combinations from massive features, and it also contributes to achieving better recommendations through a combination of tree-based model and neural network. Extensive experiments on two large public data sets are conducted on TMRM, and the results demonstrate the superiority of our proposed method over state-of-the-art solutions on performance of recommendation systems.**

***Index Terms*—recommendation systems, feature selection, multi-task learning**

I. Introduction

With the development of artificial intelligence, personalized recommendation has become an important part of intelligent networks, and it has been widely used in various practical systems. Most recommendation models work to improve their accuracy by using complex structures and large amount of features, such as Youtube's DNN [6], Wide&Deep [5]. These models can gain effective information from massive features by exerting their advantages of generalization and feature combination of neural network, so as to get good recommendation results. However, it's known that each feature has a different impact on the recommendation results, we can not know whether the model has actually learned the information of important features. As a result, we will fall into a difficult situation that although we can improve the performance of the model by adding a large number of features, when the model performance reaches its bottleneck, we can't judge whether this is the upper limit of the model, or because the model does not correctly learn the importance of different features. We have reason to believe that if we can tell the recommendation model which is an important combination of features to guide the training of the model, the recommendation result will be better.

At present, many methods have been developed for selecting important features to verify the reliability of recommendation systems, such as LIME [14], in press RSLIME [21], Anchors [15]. For the same user-item-result tuple, the important features they provide remain the same regardless of the model structure. However, these methods are post-tested for the importance of features after the recommendation model is completed. They only can test the model but can not improve its performance.

In order to make the automatic discovery of important feature combinations from massive features, and to improve the accuracy of recommendation models, a Two-stage Multi-task Recommendation Model boosted feature selection (TMRM) is innovatively proposed, which can achieve the following goals simultaneously:

1) Select important feature combinations automatically
2) Leverage the learning ability of neural network to obtain higher accuracy through multi-task learning
3) Give an important feature combination along with an accurate recommendation result, informing users of the important features that influence this recommendation.

The rest of this paper is organized as follows: Section II elaborates related work on feature selection and recommendation; Section III demonstrates the details and implemetation of the TMRM model proposed in this paper, and Section IV is the experimental analysis based on the model, which includes both accuracy analysis and feature selection analysis.

II. Related Work

Recommendation models that automatically selecting features and cross-combining features fall into three categories: tree-based model [10], [18], [20], factorization-based model [1], [4], neuron-network-based model [5]–[7], [9].

Tree-based model. Tree model such as gradient boosting decision tree(GBDT) is fused with other methods that improve generalization to draw on the merits of both models. The work [20] and [3] use GBDT instead of manually crafting cross features, and then feed the results into the embedding model. However, tree model that boosted by embedding method is difficult to make final feature selection.

2375-0197/19/$31.00 ©2019 IEEE
DOI 10.1109/ICTAI.2019.00-89

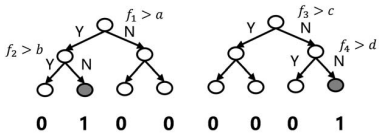

Fig. 1. Stage of feature selection.

Factorization-based model. In order to make factorization have an interpretable meaning, Behnoush Abdollahi [1] use the vector of item which is positively selected by users to filter all item factorization. Xu Chen [4] and Yongfeng Zhang [19] add phrase-level sentiment analysis to matrix factorization. However, due to computational complexity, all the above models are limited to low-order feature combination, which limits the generalization ability.

Neural-Network-based model. These models, [5]–[7], [9], use neural networks to obtain high-level features and improve the generalization and accuracy, but they all ignore the importance of giving final important cross feature learned by models explicitly, which makes them easily lose the trust of users.

We believe that a good recommendation model can not only deal with tremendous features, but also automatically extract important combinations of features behind the data. Consequently, TMRM is designed in this paper to achieve this goal. Detailed model description can be found in section III.

III. Architecture

We will describe the structure of our TMRM in detail in this section, and verify its performance in section IV.

For a clearer description, the following definitions are used in this paper: N for the number of features, M for the number of samples, $X \in R^M$ for the sample set, $x \in X$ for a single sample, $F \in R^N$ represents a feature set, and $f \in F$ represents a single feature.

A. Stage of feature selection

It is known that the importance of each feature is not equal, and those important features are the points that influence user's decisions. In order to explore the important features automatically, we adopt tree model, which selects feature and its split value with highest information gain as the split condition of tree nodes, and then the data are divided into two parts. Compared to one tree alone, multiple trees are more expressive and can better identify effective features and feature combinations. As a classic multi-tree model, Gradient Boosting Decision Tree(GBDT) prefers features with high discrimination on the overall data. Hence, GBDT is used to analyze the samples and get the split result of all samples, that is, the output of all leaf nodes in GBDT.

As shown in Fig. 1, we take two trees as an example. For sample x, x passes through the first tree of GBDT to the second. After that, it is assigned to the fourth leaf node. Then the split path of x is $(f_1 > a, f_2 > b, f_3 > c, f_4 > d)$. Assume the training target of GBDT is user preference, which is represented by CTR. And let the output of leaf node is 0 or 1, then x gets a sparse 0-1 vector 01000001 after GBDT, which can restore the split path of the sample. Then it can be used as ground truth of important features in stage 2. So we train GBDT to get vectors of all samples as feature combination set $E = \{E_1, E_2, ...E_m\}$, and each $E_i = \{e_1, e_2, ...e_p\}, i \in (0, m)$, each e takes a value of 0 or 1, representing the output of a leaf node of GBDT, and the value of p depends on the depth and number of trees.

While training GBDT to get the predicted results, we keep the activated leaf nodes of all trees as cross-features, denoted as y_2, where $y_2 \in E$.

B. Multi-task learning stage

As shown in Fig.2, the second stage of TMRM is a multi-task learning module which works to improve the recommendation accuracy by learning the obtained data rules in stage 1 and user preferences. This stage is designed to achieve this goal as learning the cross features is beneficial to learning users' preference.

The input of this module adds ID features which include User-ID and Item-ID (represented by f_u and f_v) to all features used in the first stage. And this module is divided into pre-processing layer, hidden layer, and output layer. One of this module's tasks is to predict whether the item will be clicked. It is a binary classification task, assume $\hat{y}_1 \in \{0, 1\}$ is the prediction of CTR, then binary-classifier h is $X \to \hat{y}_1$. Another task is to learn the important cross-features selected by stage 1, which is equivalent to a multi-label classification task. As defined in Section III-A, the important cross-feature set is $E = \{E_1, E_2, ..E_m\}$, where $E_i = \{e_1, e_2, ...e_p\}$ is output of the leaf node in stage 1, then the multi-classifier I is $X \to \hat{y}_2$, assigning E_i to each instance $x_i \in X$, so the output of classifier I is a vector $\hat{y}_2 = (I_1(x), I_2(x), ..., I_p(x))$. Apart from the output layer, these two tasks share structure and parameters of each other.

Pre-processing layer densifies features of sample x. Drawing on the idea of FFM [12], the input was divided into multiple fields. As shown in Fig. 2, only f_2 is 1-value in field 1, then each neuron in the corresponding pre-processing layer is activated by only one non-zero value, so we get a vector $v_2 * f_2$ to represent filed 1 where v_2 is $(v_{21}, v_{22}, ..v_{2k})$. Hidden layer is divided into linear part, low-order-feature part and high-order-feature part. The input of linear part is all the features of the original input layer. And input of the rest two parts are the same which are the output $(v_1 f_1, v_2 f_2, ..v_n f_n)$ of the pre-processing layer. These two parts do linear weighting and second-order combination of features, which can be expressed as:

$$y_{linear} = w_0 + \sum_{i=1}^{n} w_i f_i \tag{1}$$

$$y_{low\_order} = \sum_{i} \sum_{j} v_i v_j f_i f_j \tag{2}$$

The high-order feature part is a 3-layer fully connected network, and the features are deeply cross-combined. We let

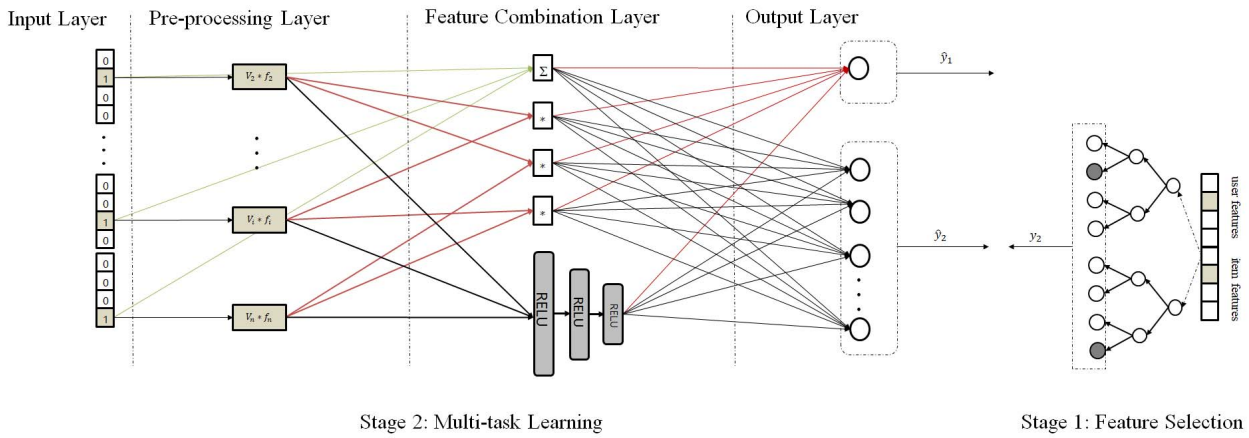

Fig. 2. Architecture of TMRM, stage 1 on the right, used to filter important feature combinations, stage 2 on the left, using the important features selected in stage 1 for multi-task learning.

L denote numbers of hidden layer of this part, $l \in L$,and H^l denotes the l-th output, then the forward process is:

$$H^{l+1} = ReLU(W^l H^l + B^l) \tag{3}$$

$$y_{deep} = WH^{L+1} + B^{L+1} \tag{4}$$

where the output is y_{deep} . Output layer is divided into two parts to get the results of recommendation($\hat{y}_1$) and important cross-features($\hat{y}_2$).

$$\hat{y}_1 = \sigma(y_{linear} + y_{low\_order} + y_{deep}) \tag{5}$$

$$\hat{y}_2 = \sigma(Re^T(y_{linear}, y_{low\_order}, y_{deep}) + b) \tag{6}$$

where $y_{low\_order} \in R^k$, and y_{deep} is obtained by formula.4. If $y\_deep \in R^s$, then $Re \in R^{(1+k+s)*p}$, $\hat{y}_2 \in R^p$, where k is the dimension of the vector v mentioned above, and s is the number of nodes in the last layer of the deep part.

C. learning

For the learning of recommendation results, we regard it as a binary classification task, and the loss function is

$$L_1 = \sum_i -y_1 log\hat{y}_1 - (1-y_1)log(1-\hat{y}_1) \tag{7}$$

where y_1 stands for ground truth.

For the learning of cross features, we consider it as multi-label classification tasks, and we dealt learning of multiple feature combinations as independent to simplify the calculation. Then the loss function is

$$\begin{aligned} L_2 &= \sum_i logP(y_2|I) \\ &= \sum_i log(\Pi_{j=1}^{j=p} P(y_{2_j}|I_j)) \\ &= \sum_i \sum_j [-y_{2_j} log\hat{y}_{2_j} - (1-y_{2_j})log(1-\hat{y}_{2_j})] \end{aligned} \tag{8}$$

where i represents the i-th sample, and j represents the j-th feature. For the first stage, the optimization goal is L_1, and loss of the second stage is combination of L_1 and L_2, which is denoted by L. Two methods are used in this paper. The first is to learn the optimal weights to linearly weight multiple tasks to get L_{OW} inspired by [16], which is also the general practice of multi-task learning. The other is to introduce homogeneity uncertainty inspired by [13] to get L_{HU}, which suggests multi-task learning is likely to predominate certain tasks, making other tasks less adequately optimized. Therefore, the weight problem of different tasks is solved from the perspective of task dependency uncertainty based on their schemes. Thus we get

$$L_{OW} = w_1 L_1 + w_2 L_2 + \lambda^2 \tag{9}$$

$$L_{HU} = [\frac{1}{2\lambda_1^2} L_1 + log(\lambda_1^2)] + [\frac{1}{2\lambda_2^2} L_2 + log(\lambda_2^2)] \tag{10}$$

where w_1,w_2 represent the weights of recommendation task and feature selection task respectively, which are automatically trained.λ_1,λ_2 are two scalar of observation noises to adjust multi-task weights dynamically.

We use the two loss functions mentioned above to train our TMRM models, which are described as TMRM-OW and TMRM-HU. We verify their performance in section IV.

IV. Experiments

The main contribution in this paper is to improve the recommendation performance while giving important feature combinations. We will answer the following three questions through a series of experiments: 1) RQ1: Can TMRM achieve higher accuracy when compared with state-of-the-art models? 2) RQ2: Can TMRM be effective in feature selection? 3) RQ3: How do different hyper-parameters affect TMRM?

A. Dataset

We selected two public data sets to validate TMRM: e-commerce data([1] [11], called Retail) and movie data (a com-

[1]https://www.kaggle.com/retailrocket/ecommerce-dataset

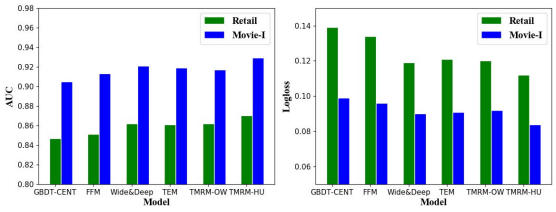

Fig. 3. Performance comparison of AUC and Logloss in different models.

bination of Movielens-10M[2] and IMDB[3] [2], called Movie-I). The detailed information about these two data sets is in Table.I.

TABLE I
STATISTICS OF THE DATASETS

Dataset	#User	#Item	#Features	Interaction
Retail	11719	12511	1769	22457
Movie-I	2113	10197	112695	855598

B. Evaluation Metric

For recommendation, we randomly select 5 negative samples for one positive user-item sample. For feature selection, we adopt the idea of NCE [8] to update parameters. In order to evaluate the prediction results of the model, we choose AUC and logloss as evaluation indicators of recommendation accuracy. In addition, evaluation of the cross features selection will be analyzed as case study in this paper.

C. Model Comparison

We compare our proposed TMRM with the following state-of-the-art models to verify its performance. GBDT-CENT [20], FFM [12], Wide&deep [5] and TEM [18], which are classic models of automatic feature selection.

D. Performance of Recommendation (RQ1)

Fig.3 exhibits the performance of different models on two data sets. The performance of our TMRM-OU model exceeds that of FFM and GBDT-CENT, but it can't be better than wide&deep and TEM. Because the simple linear weighting of multi-task model makes the model tend to favor one task and neglect the learning of another task. This makes multi-task learning dominated by single task, and it can not achieve the effect of joint training. But our TMRM-HU model can over-perform all above models in recommendation performance, reflecting that the introduction of homogeneity uncertainty improves the learning of multi-task.

[2]http://www.grouplens.org
[3]http://www.imdb.com

E. Case Study (RQ2)

In order to illustrate the accuracy and validity of the feature combinations selected by TMRM, we conducted the following case studies only on Movie-I because the fields of Retails are all hashed, it is not convenient to intuitively obtain the true meaning.

For each participant, we provide the movies that a user in dataset has seen and the tags he/she has given for the participant. They need to infer the user's preferences and the features of the user's attention to movies. Then, we will provide these participants with the recommendation results from our TMRM and the important features for each recommended movie. They will give an evaluation through our well-designed questionnaire.

In order to make the questionnaire more objective and reliable, three questions are designed as follows:
Q1: Are you satisfied with this recommendation? Q2: Is the features of the movie given in this recommendation correct? Q3: Are you interested in the movie's features given in this recommendation? According to [17], we design a 5-score answer, where 1 represents strongly negative and 5 indicates strongly positive. One point to note is that we select Wide&Deep as the representative from other models that cannot give feature selection but have high accuracy, give feature selection by randomly selecting the movie tag(denoted as Random) and by selecting the popular tag of the movie(denote as Hot).

TABLE II
RESULT ANALYSIS OF CASE STUDY

MeanValue / Question / Strategy	Q1	Q2	Q3
Random	3.641	2.257	2.143
Hot	3.641	3.142	3.011
TEM	3.940	3.704	3.546
TMRM-HU	3.978	3.712	3.500

After filtering some invalid questionnaires, we received a total of 100 answers. As TableII shows, we use average to represent the final result of a model and its feature selection strategy. It shows that TMRM can indeed learn important and effective features.

F. Hyper-parameter Studies (RQ3)

We selected several typical hyper-parameters to study their impact on model performance. Fig.4 shows the effect of number and depth of trees on AUC, which mainly affect the results of our model's first stage, section III-A. Fig.5 shows the effect of dropout and embedding size on our model's Logloss, which are mainly for multi-task learning stage.

V. CONSLUSION

In this paper, a Two-stage Multi-task Recommendation Model, TMRM, is proposed. With a two-stage process for samples and a multi-task learning approach, TMRM can achieve excellent results in terms of accuracy. In contrast with

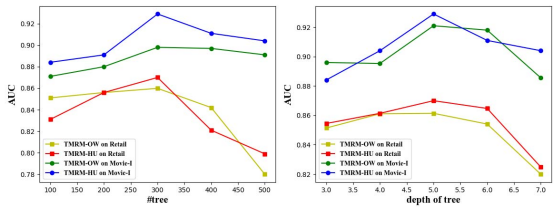

Fig. 4. Performance comparison of AUC w.r.t.the tree number and the depth of tree

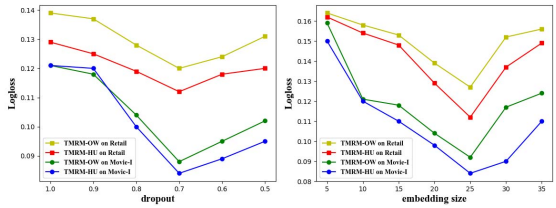

Fig. 5. Performance comparison of Logloss w.r.t. dropout values and the embedding size.

other recommendation models, TMRM considers the influence of important feature combinations, and uses the tree model to extract important cross-features in advance to guide the training of neural networks. Based on TMRM, we provide personalized recommendations and give item's features in line with user's interests. Comprehensive experiments results validate the performance and advantage of TMRM.

ACKNOWLEDGMENT

This research work was supported by the National Natural Science Foundation of China under Grant No.61802029, the fundamental Research for the Central Universities No.500418800.

REFERENCES

[1] Behnoush Abdollahi and Olfa Nasraoui. Using explainability for constrained matrix factorization. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*, pages 79–83, 2017.

[2] Iv'an Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA, 2011. ACM.

[3] Tianqi Chen, Linpeng Tang, Qin Liu, Diyi Yang, Saining Xie, Xuezhi Cao, Chunyang Wu, Enpeng Yao, Zhengyang Liu, Zhansheng Jiang, et al. Combining factorization model and additive forest for collaborative followee recommendation. *KDD CUP*, 2012.

[4] Xu Chen, Zheng Qin, Yongfeng Zhang, and Tao Xu. Learning to rank features for recommendation over multiple categories. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*, pages 305–314, 2016.

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, DLRS@RecSys 2016, Boston, MA, USA, September 15, 2016*, pages 7–10, 2016.

[6] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 191–198, 2016.

[7] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: A factorization-machine based neural network for ctr prediction. *CoRR*, abs/1703.04247, 2017.

[8] Michael Gutmann and Aapo Hyvarinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, pages 297–304, 2010.

[9] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 355–364, 2017.

[10] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Qui nonero Candela. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, ADKDD 2014, August 24, 2014, New York City, New York, USA*, pages 5:1–5:9, 2014.

[11] Bal'azs Hidasi and Alexandros Karatzoglou. Recurrent neural networks with top-k gains for session-based recommendations. *CoRR*, abs/1706.03847, 2017.

[12] Yuchin Juan, Damien Lefortier, and Olivier Chapelle. Field-aware factorization machines in a real-world online advertising system. In *Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017*, pages 680–688, 2017.

[13] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7482–7491, 2018.

[14] Marco T'ulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?:explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.

[15] Marco T'ulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1527–1535, 2018.

[16] Jonas Uhrig, Marius Cordts, Uwe Franke, and Thomas Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. In *Pattern Recognition - 38th German Conference, GCPR 2016, Hannover, Germany, September 12-15, 2016, Proceedings*, pages 14–25, 2016.

[17] Nan Wang, Hongning Wang, Yiling Jia, and Yue Yin. Explainable recommendation via multi-task learning in opinionated text data. In *The 41st International ACM SIGIR Conference on Research Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 165–174, 2018.

[18] Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. Tem: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, pages 1543–1552, 2018.

[19] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast , QLD, Australia - July 06 - 11, 2014*, pages 83–92, 2014.

[20] Qian Zhao, Yue Shi, and Liangjie Hong. Gb-cent: Gradient boosted categorical embedding and numerical trees. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 1311–1319, 2017.

[21] Fan Zhu, Min Jiang, Yiming Qiu, Chenglong Sun, Zeming Zhang, and Min Wang. Rslime:an efficient feature importance analysis approach for industrial recommendation systems. 2019, in press.