4-track Recommending Models along with GBDT Ranking

LI LIANG, Shanghai University of Finance and Economics, China
YONGGUI LUO, Peking University, China
HUILING CAI, Tongji University, China
KAICUN WU, Southeast University, China

CCS Concepts: • **Information systems → Collaborative filtering**; • **Computing methodologies → Boosting**.

Additional Key Words and Phrases: item2vec, bipartite networks, CatBoost

1 INTRODUCTION

This report is aim to explain the 16th solution of KDD Cup 2020 Challenges for Modern E-Commerce Platform: Debiasing.

2 METHOD

2.1 Item-Based CF

CF (collaborative filtering) is a classic recommending solution which works by building a user-item co-occurrence matrix. Traditional CF algorithm is a user-based nearest neighbour algorithm which searches for neighbour users for the target user[3]. However, with millions of user and relatively sparse items interacted by each user, this method has the defects of high computational cost and poor customization.

Item-Based CF bypasses those problems by a two-step process. First, it analyzes the user-item matrix to identify relations between items by similarity calculation. Second, based on the calculation result and history interactions of the target user, it cumulatively computes the prediction scores for items haven't been interacted with yet[5]. The top-N recommendations are given by sorting the prediction scores.

2.2 Item2Vec

Skip-gram with Negative Sampling (SGNS), also known as Word2Vec, is a neural word embedding method[4]. The method aims at finding words representation that captures the implicit relations between words in a corpus. It was shown to provide state-of-the-art results on various linguistics tasks. For details, SGNS aims at maximizing the following term:

$$\frac{1}{K}\sum_{i=1}^{K}\sum_{j\neq i}^{K}\log p\left(w_j|w_i\right), \tag{1}$$

$$p\left(w_j|w_i\right)=\frac{\exp\left(u_i^T v_j\right)}{\sum_{k\in I_V}\exp\left(u_i^T v_k^T\right)}, \tag{2}$$

where K is the length of a sequence of words, c is the context window size and w_i stands for a word at position i.

Inspired by the achievement in the field of NLP, Item2Vec which incorporates embedding ideas into item-based CF was developed for inferring item-to-item latent relations even when the users' information is not available. Each user's

Authors' addresses: Li Liang, Shanghai University of Finance and Economics, China; YongGui Luo, Peking University, China; HuiLing Cai, Tongji University, China; KaiCun Wu, Southeast University, China.

item interaction history can be seen as a "sentence" and each item is a "word". Each pair of items sharing the same set is treated as a positive example. Item2Vec replaces the softmax function from Eq.(2) with:

$$p\left(w_j|w_i\right) = \sigma\left(u_i^T v_j\right) \prod_{k=1}^{N} \sigma\left(-u_i^T v_k\right). \tag{3}$$

where N is a parameter that determines the number of negative examples to be drawn from a positive example, $\sigma(x) = 1/1 + \exp(-x)$, $u_i \in U\ (\subset \mathbb{R}^m)$ and $v_i \in V\ (\subset \mathbb{R}^m)$ are latent vectors that correspond to the target and context representations for the word $w_i \in W$, respectively, $I_W \triangleq \{1, \ldots, |W|\}$[1].

2.3 Bipartite Network based Inference

Bipartite Network (BiNe) is a subset of complex networks with bipartite graph structure which only allows the connections between nodes from two disjoint and independent sets. Many real life connections can be modeled as a bipartite network, for example, the relationship between readers and books can be generalized as a BiNe. However, with the traditional one-mode projection method, the edges between too nodes are unweighted which inevitably leads to information loss.

BiNe was applied to personalized recommendation for the first time in 2007 along with a new projection method[6]. With this new resource allocation based weighting projection method we can not only overcome the information loss but also naturally give out recommendations. The weighted result matrix has two precious properties. First, the weighted matrix is not symmetrical and the node having larger degree in the BiNe generally assigns smaller weights to its incident edges. Second, the diagonal element in the weighted matrix is positive, which makes the weighted one-mode projection more informative[6].

2.4 CatBoost

Catboost is a machine-learning algorithm based on Gradient Boosting Decision Tree (GBDT). It is good at handling categorical features and does well in reducing over-fitting. Decision tree is a tree structure in which each internal node represents a judgment on an attribute while each branch represents a judgment on a output, and each leaf node represents a classification. Gradient Boosting is an ensemble technique that combines weaker models (base predictors) via a greedy procedure that corresponds to gradient descent in a function space. Unlike bagging ensemble, all base predictors are strongly related in gradient boosting.

Catboost has some features that are different from other GBDT models like LightGBM and XGBoost. First, it uses symmetric tree as base predictor to help avoid over-fitting, increase reliability. Second, it handles categorical features in a special way as it calculates the frequency of occurrence of a category to generate new numerical features and make combinations between features. When constructing the first node, only one feature is selected. To the second node, it will try to combine the existing node and any others categorical features and choose the best of the pairs. Third, it presents an solution to avoid the "biased point-wise gradient estimates". Model M_i is built using all training sets (except for the i-th data), and then a correction tree ΔM is built using the 1st to i-1st data and accumulates to the original model M_i[2].

3 RESULTS

We trained 4 models for each phase separately for recommending part, and trained a classification model based on phase 1-5's data for ranking part. Top-500 items were recalled on every track of recommending. The last-click records were filter out for all users as the offline test set.

3.1 Recommending

In our implementation, altogether two different Item-Based CF models were trained as two of four tracks. We used cosine-based similarity to do basic calculations. Also, four different penalties were added to active users for the sake of debiasing:

- penalty to distance, the further the less relevant
- penalty to length of item list, the longer the less valuable
- penalty to negative orientation
- penalty to hot items

The difference between those two tracks was that one of them only computed with 20 items window and the other didn't limit the window. Also, there were some slight numerical difference on penalty computation equations.

Table 1. Specified parameters of Word2Vec

Parameters	Description	Value
iter	Number of iterations (epochs) over the corpus.	60
alpha	The initial learning rate.	0.025
seed	Seed for the random number generator	42
min_count	Ignores all words with total frequency lower than this.	1
window	Maximum distance between the current and predicted word within a sentence.	99999
sg	Training algorithm: 1 for skip-gram; otherwise CBOW.	1
hs	If 1, hierarchical softmax will be used for model training. If 0, and negative is non-zero, negative sampling will be used.	0
size	Dimensionality of the word vectors.	128

We applied Item2Vec using package *gensim.Word2Vec* to both train and test dataset as the third track model. All custom parameters are shown in Table 1. During fine tuning, we found the *alpha* parameter was the key for expecting a good performance. The forth track was bipartite network based inference. We incorporated ideas from BiNe projection to generate an asymmetry weight dictionary between items. The weight w_{ij} can be interpreted as the strength of item i to a user provided he has clicked item j. According to the first property mentioned in Mehod part, w_{ij} is unlikely equal to w_{ji} which is consistent with intuition and hot items will be recommended with lower probability which is pertinent to the goal of debiasing.

3.2 Ranking

By merging all recall results as a dataframe, the ranking task was converted to a binary classification problem. Positive label identified the predicted item was equal to true item. Since the negative samples were overly excessive, we subsampled to 50 items. Feature engineering can be summarized to two parts. The first part was the rank and score from the preceding 4-track recommendations. The second part included numerical combination of feature vectors and similarity of text and image vectors. We filtered out the target user's recalled item and the user's last one, last two and last three click item and calculated similarity though those vectors.

Table 2. Specified Parameters for CatBoost

Parameter	Description	Value
iterations	maximum number of base tree predictors	20
learning_rate	control the speed for parameters update	0.1
loss_function	Specific loss function to minimize	'logloss'
class_weights	class weights for samples	[1, 20]
custom_metric	Metric values to output during training	'F1'

Parameters of the CatBoost model are shown in Table 2. By comparing the loss function curves and F1 score for training set data and test set data, we set iterations=20 for early-stopping. With class_weights=[1,20], model can get a better F1 score in the case of unbalanced positive and negative samples.

4 CONCLUSION AND DISCUSSION

The e-commerce industry plays an increasingly more important role in the global market with an estimated 6.54 trillion dollars global revenues in 2022. To achieve better customer experience and higher sale revenues, companies implement recommender systems. A slight improve in recommendations can boost revenues greatly, that's why we pursue a fair recommender system with credible performance. Our team's solution is a combination of multi-track recommending and GBDT ranking, which is simple but practical. It achieved relatively high performance on *NDCG@50* and *hitrate@50* metrics. Actually, we have tried other models like item-item CF, hybrid models and Matrix Factorization etc. Unfortunately, results of this trails were far from satisfying. Due to the limitation of computation resources and time distribution, we can imagine there are still opportunities to improve.

REFERENCES

[1] Oren Barkan and Noam Koenigstein. 2016. Item2vec: neural item embedding for collaborative filtering. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 1–6.

[2] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. 2018. CatBoost: gradient boosting with categorical features support. *arXiv: Learning* (2018).

[3] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[4] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[5] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[6] Tao Zhou, Jie Ren, Matúš Medo, and Yi-Cheng Zhang. 2007. Bipartite network projection and personal recommendation. *Physical review E* 76, 4 (2007), 046115.