Session Recommendations With Song Features Based On Transformer

Shuyi Li

School of Computer Science and Engineering, Guangzhou Institute of Science and Technology, Guangzhou City, Guangdong Province, China

e-mail: lsy782280720@163.com

Abstract. With the advent of the information age and the explosive growth of data, it is difficult for people to find the content they are interested in, and the recommendation system came into being. In recent years, the session recommendation in the recommendation system has attracted the attention of many people, which is obviously different from the traditional recommendation. The traditional recommendation, such as collaborative filtering, is to find similar users or similar items for recommendation, while the session recommendation is to recommend according to the interaction behavior between users and items, so that dynamic recommendation can be made according to the changes in users' interests. In this article, we first proposed to use the transformer based method to learn the historical behavior of users and the label characteristics of each song, then search the item set that is most similar to the last item in the session through the itemknn method, and mix it proportionally to form the final recommendation list. At last, a large number of experiments are carried out to prove the superiority of our method.

Keywords: conversation recommendation, transformer, long-term and short-term recommendation

1 Introduction

Today's society is deeply influenced by recommendation. When you want to watch news, the recommendation system will introduce the latest news; when you brush short videos, the recommendation system will recommend the short videos that we are most interested in; when you want to shop, the recommendation system will pick out the items we most want, so the recommendation system has gained more and more attention. It can give people a better experience and get the content they are interested in more quickly. For example, there are a lot of songs in the music database, people can not listen to all the music, a lot of music may become "long tail songs" because of heat, age and other reasons, but these are likely to be the user's favorite music. Recommendation system can solve this problem.

The items that the user clicks on are usually the ones that the user is interested in. Sachdeva et al. [1] use the "Twin tower model" to convert the item information into

C. Chen et al. (eds.), *Proceedings of the 3rd International Conference on Digital Economy and Computer Application (DECA 2023)*, Atlantis Highlights in Computer Sciences 17,
https://doi.org/10.2991/978-94-6463-304-7_9

embeddeding vector and the user item click sequence into the embedding vector. On the other hand, they will learn the user's behavior sequence through bigru and the attention mechanism to obtain the user's short-term preferences. However, the problem with this approach is that it can only learn the user's short-term preferences, and when the user's click sequence is too long, there is no way to capture the user's previous preferences. Jannach et al. [2] tried to combine GRU and KNN, using GRU to obtain users' long-term preferences and KNN to obtain users' short-term preferences, so as to capture users' long-term and short-term preferences. However, GRU still could not completely capture users' long-term preferences due to the obvious gradient disappearance problem.

Our work uses a three-tower model. First, we use the transfomer model to capture long-term preferences, which can solve the problem of gradient loss in bigru, and then we use transformer to learn the label characteristics of each song to improve the recommendation. The hidden vectors in transformer are merged and sent to MLP. Finally, softmax is used to generate the recommendation list. In addition, the last item in the sessilon is obtained, converted into a vector, itemKNN is used to find the most similar items, and finally the final recommendation list is generated according to the proportion. The main contributions of this paper are as follows: (1) A new model architecture is proposed, which uses transformer model to learn long-term user preferences and learn the label characteristics of each item. (2) Obtain users' short-term preferences through itemknn algorithm. (3) A large number of experiments have been done to prove that our method is more advantageous.

2 Related work

2.1 Session-based recommendations

Markov chain model. Traditional sequential model recommendations include the use of Markov chain model recommendations, and Markov chain recommendations include traditional Markov chain-based recommendations, which were proposed by Yap[3] et al., to observe conversion probabilities by displaying features. There is also latent Markov embedding-based method, which maps the feature to Euclidean space through the embedding method, and then obtains the next item that may be clicked by calculating the Euclidean distance [4]. But this approach captures short-term preferences, not long-term ones.

Embedding. Some work usually converts the information about the interaction between the user and the item into the embedding vector, and then feeds the embedding vector into the neural network to calculate the score to predict the next item [5]. Other work directly uses the embedding vector to calculate the score through the Euclidean distance, predicting the next item to be clicked. [6]

2.2 Hybrid Recommendation Method

In the Wide&Deep model proposed by Google [12], the "wide" part enhances the "memory ability" of the model, and the "deep part" enhances the generalization ability

of the model. The DeepFM model [13], proposed by Harbin Institute of Technology in collaboration with Huawei, replaces the wide part of the Wide & Deep model with the FM model to enhance the capability of shallow network partial feature combination.

2.3 Sequence recommendation based on deep learning

RNN-based sequence recommendation. Using RNNS to learn the historical sequence of a user's behavior and then predict what items are likely to be clicked next is a very efficient method. In addition to using RNN, Wu et al. [7] used LSTM[8] for sequence recommendation, which can effectively learn more long-term user preferences and solve the problem of gradient disappearance to a certain extent. Hidasi et al. [9] obtained long-term preferences by using GRU, which can reduce the number of parameters and improve the training speed. However, the biggest disadvantage of using recurrent neural networks is that it may learn noise, such as the user mistakenly clicks on the product, but the model treats it as a valid click, causing bias.

CNN-based sequence recommendation. Tang et al[10] use cnn to make sequence recommendation by converting all the interactive information into the embedding vector, then merging these embedding vectors into a matrix, which is similar to an image, and learning relevant sequence information through convolution kernel, which can enhance robustness to a certain extent. However, long-term preferences cannot be captured due to matrix size limitations.

3 Our approach

Our model is mainly composed of three parts. The first part is to obtain long-term preferences of users using transformer, encode the user click sequence using onehot, and then feed the embedding layer into the embedding vector. Use transformer to learn the behavior sequence, Part 2 is similar to Part 1, except that Part 2 feeds the labels of each piece of music, which will be detailed below. In the third part, the last item vector is obtained by using itemKNN method, and the most similar item is obtained by using Euclidean distance. Finally, we combine the results from these two modules to produce the final recommendation list.

3.1 Use transfomer model

S={s1,s2,…,si-1}, represents the sequence in which the user clicks on the song. T={ti1,ti2…,tij} represents the label characteristics of each song. The goal is to predict si. As shown in Figure 1. Use transformer to learn long-term user preferences.

$$\mathbf{s}_i^{'} = E_1 * s_i \tag{1}$$

Si represents the one-hot vector of the song, and E1 is a embedding layer, mapping the one-hot vector of the song to the embedding vector.

$$\mathbf{t}_i^{'j} = E_2 * t_i^j \tag{2}$$

Representing the Jth feature of song i, E2 is a embedding layer that maps the one-hot vector of the song label to the embedding vector.

$$t_i^{'} = \frac{1}{n_i} \sum_{j=1}^{n_i} t_i^{'j} \tag{3}$$

n_i is the number of songs, because each song has several labels, here all the label vectors are added and averaged to form a vector.

Next we feed the embedding vector into the transformer layer. Next, we obtain the implicit vector H_1 of transformer, which contains the entire sequence information. For the second part, the label vector of the song is the same as in the first part, and then the two implicit vectors are concatenated.

$$H = concat(H_1 + H_2) \tag{4}$$

H1 represents the item sequence information of transformer[14]. We extract the last layer of hidden vector, which contains all the item information clicked by users. H2 is also the last layer of hidden vector, which contains the sequence information of all users' click item label features in the sequence, and H represents a larger hidden layer vector formed by concatenating H1 and H2 two hidden vectors.

$$C^{'} = \mathrm{Re}\,LU(W_1 H + b_1) \tag{5}$$

Here, we use the activation function relu, where C 'represents a vector of smaller dimensions in order to accelerate convergence by reducing training time.

$$O = soft\max(C') \tag{6}$$

Finally, through the softmax function, output a probability.Output a probability, with the highest probability representing the item most likely to be clicked next.

3.2 Use itemKNN algorithm

The itemKNN algorithm is used mainly because the last song clicked by the user is likely to be the type of song that the user most wants to listen to. The implementation method is as follows

(1) Walk through each session and get the last item of each click session;

(2) Construct the item correlation matrix according to the item id in the item list;

(3) KNN, using Euclidean distance to find the most relevant items;

(4) Return to the song list.

We combine the songs selected by the two strategies in a certain proportion to generate the final recommendation result.

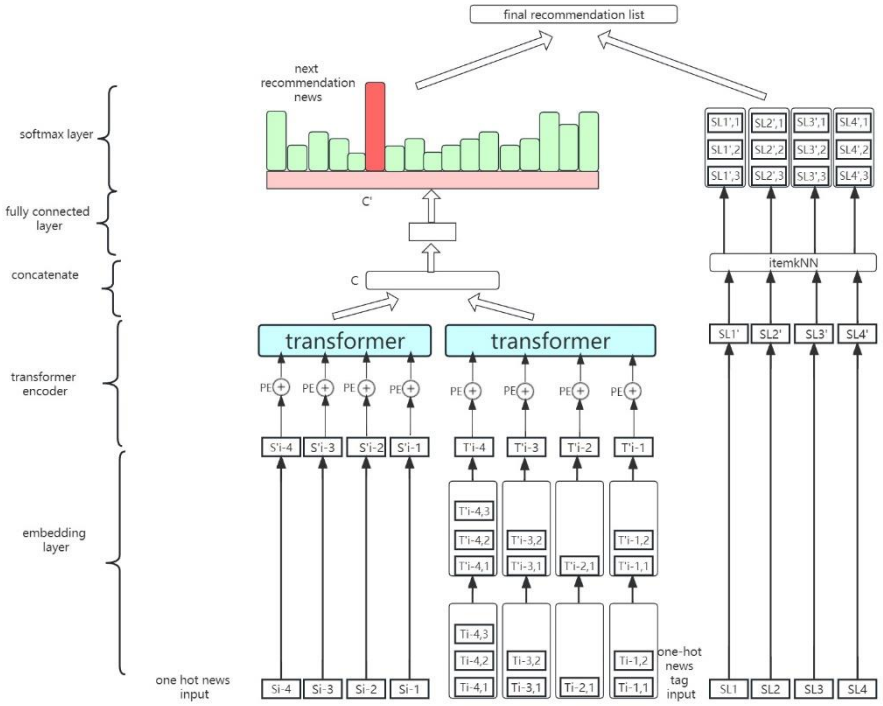

Fig. 1. STIBRS model

4 Experiment

4.1 Dataset

The dataset we used is the Last.fm dataset. The data set contains user id, song name, artist name, and time stamp. We obtained 6 months 'data for experiment, taking 70% of the data as the training set and 30% as the test set. In addition, we preprocessed the data set and deleted sessions with fewer than 5 songs.See Table 1 for information about the dataset.

4.2 Baseline

BPR-MF[10]: Bayesian personalized sorting model based on matrix decomposition.

Table 1. Last fmBasic data set information

Description	Value
Total Logs	3553321
Total Users	759

Total Sessions	110410
Total Unique Songs	386046
Total Unique Tags	487844
Average Songs Per Session	32.18
Average logs per user	4681.58

RNN [11]: In this method, a sequence of items is fed into a recurrent neural network to try to predict the next item.

Subsession Based Recommender System[5]: In this approach, short-term user preferences are recommended by using song tags.

STABR: The interactive sequence is learned by using the method of bidirectional GRU plus attention mechanism.

GRU4REC: Using GRU, a combination of session-based KNN.

Table 2. Result

Model	K=10	K=20	K=30	K=40	K=50
BPR-MF	7.34	8.13	8.56	8.98	9.27
SSCF	13.69	17.12	19.66	21.30	22.34
RNN	14.42	16.26	16.74	17.09	17.38
SBRS	19.15	26.14	28.83	30.35	31.40
STABR	28.95	30.85	31.90	32.65	34.26
GRU4REC	29.12	31.26	32.28	33.49	35.20
OUR	**30.13**	**33.41**	**33.51**	**35.17**	**36.20**

4.3 Training&Testing

We use several optimization algorithms, including stochastic gradient descent algorithm, Adagrad method, and Adam method. The comparison results are shown in Table 3. We chose the adam algorithm.

Table 3. Optimizer selection

Optimizer	K=10	K=20	K=30	K=40	K=50
Sgd	28.31	29.52	30.71	31.24	32.43
Adagrad	27.24	28.24	29.22	30.45	32.87
Adam	**30.13**	**33.41**	**33.51**	**35.17**	**36.20**

The learning rate of each training model is set at 0.005, which is also the result of comparison through a large number of experiments, Including 0.001, 0.003, 0.005, 0.007, 0.009. The comparison experiment results are shown in Table 4. We use a stochastic gradient descent algorithm, the learning rate of each training model is set to 0.01, the batch size is 32, the length of the label embedding is 25, and the song embedding length is 50. The intermediate MLP is (128,64), (64.32) respectively, and the final output size is 50. dropout super parameter p is set to 0.1. The number of items generated by tramsformer and the number of items adopted by itemknn are mixed in a variety of

ratios. 9:1, 8:2, 7:3, 6:4, 5:5, and 8:2 have the best effect. We trained our model on a single 4070 GPU using PYTORCH, a deep learning framework.

Table 4. learning rate choose result

Learning rate	K=10	K=20	K=30	K=40	K=50
0.001	29.13	33.23	33.30	34.98	35.88
0.003	30.01	33.19	33.12	34.68	35.44
0.005	**30.13**	**33.41**	**33.51**	**35.17**	**36.20**
0.007	29.01	32.19	32.94	34.48	35.11
0.009	29.14	31.42	32.56	33.47	34.14
0.001	29.13	33.23	33.30	34.98	35.88

The evaluation metric we used is HitRatio@k[4], where k is the number of songs in the predicted set.

The final results are shown in Table 2, and you can see that our model performs better than the other baseline models.

5 Summary

In our work, we propose STIBRS model, learn sequences with transformer model, enhance the "memory ability" of the model with itemknn model, and capture users' long-term and short-term preferences. By comparing with other baseline methods, we can find that our model has better recommendation effect. Looking forward to the next work, session recommendation is a very important part of the recommendation system. Our model has been able to capture users' long-term and short-term preferences. Next, we hope to use simple feature crossing methods to improve itemKNN, such as factorizer, so as to capture users' short-term preferences more effectively.

References

1. N Sachdeva,K Cupta,V Pudi.Attentive Neural Architecture Incorporating Song Features For Music Recommendation.In Proceeding of Twelfth ACM Conference on Recommender System(RecSys 18).ACM, New York, NY, USA,5pages.
2. Dietmar Jannach and Malte Ludewig. 2017. When Recurrent Neural Networks meet the Neighborhood for Session-Based Recommendation. In Proceedings of RecSys'17, August 27–31, 2017, Como, Italy, , 5 pages.
3. Ghim-Eng Yap, Xiao-Li Li, and Philip Yu. Effective next-items recommendation via personalized sequentialpattern mining. In Database Systems for Advanced Applications, pages 48–64, 2012.
4. Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, and Yeow Meng Chee. Personalized ranking metric embedding for next new poi recommendation. In Proceedings of the 24th International Joint Conference on Artificial Intelligence, pages 2069–2075, 2015.

5. Shoujin Wang, Liang Hu, Longbing Cao, Xiaoshui Huang, Defu Lian, andWei Liu. Attentionbased transactional context embedding for next-item recommendation. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pages 2532–2539, 2018.
6. Ruining He, Wang-Cheng Kang, and Julian McAuley. Translation-based recommendation: A scalable method for modeling sequential behavior. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, pages 5264–5268, 2018.
7. Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. Recurrent recommendernetworks. In Proceedings of the 10th ACM International Conference on Web Search and Data Mining, pages 495–503, 2017.
8. Bal´azs Hidasi, Alexandros Karatzoglou and et al. Session-based recommendations with recurrent neural networks. In Proceedings of the 4th International Conference on Learning Representations, pages 1–10, 2016.
9. Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In Proceedings of the 11th ACM International Conference on Web Search and Data Mining, pages 565–573, 2018.
10. S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme: Bpr: Bayesian personalized ranking from implicit feedback. In Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence,pp.452-461.AUAI Press,2009.
11. R. Devooght and H. Bersini.: Long and Short-Term Recommendations with Recurrent Neural Networks. Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization(2017),pp.13-21.
12. Cheng,Heng-Tze,et al.Wide & Deep learning for recommender system.Processings of the 1st workshop on deep learning for recommender systems.ACM, 2016.
13. Guo,Huifeng, et al.DeepFM:a factorization-machine based neural network for CTR prediction.arXiv preprint arXiv:1703.04247(2017).
14. Vaswani, Ashish, et al. "Attention is all you need."Advances in Neural Information Processing Systems. 2017