Reading Summary for Dynamic Graph Neural Networks for Sequential Recommendation

@Di Mo

This paper addressed the problem that ignorance of the information between different user sequence. For sequential recommendation, methods have explored the users' historical preference, however, different sequence might have some similarities about the users. and sequences. It will helpful to take those information to training. The researchers proposed a method Dynamic Graph Neural Network for Sequential Recommendation (DGSR) to address this problem. Different user sequences are combined by a dynamic graph structure. A Dynamic Graph Recommendation Network is designed to get features of user's preferences.

DGSR

The DGSR can be divided into 4 parts: input of user sequence, dynamic graph construction, dynamic graph recommendation networks and prediction layer. The user sequences are used to construct the dynamic graph construction. The dynamic graph consists users and items as nodes. The graph contains time and order messages. $\mathcal{G} = \{(u, i, t, o_u^i, o_i^u) | u \in U, i \in V\}$ where o_u^i denotes the order of item i in user u's interaction history and o_i^u denotes the order of user u in item i's interaction users. For the construction process, I am curious that whether the user items in a sequences is unique. How to deal with the replication items. Besides this problem, there is another problem is that constructing graph in this way will result in exponential increase of size. Addressing this size growing problem, the user use a sub-graph method to effectively solve this problem. After sub-graph sampling, there would be a sub-graph to be delivered. $\mathcal{G}_u^m(t_k)$ is the out put graph containing the nodes of the sequence S_u (the users sequence) and its relevant sequences. This Graph contains item to user links and user to item links. Based on input of the sub-graph $\mathcal{G}_u^m(t_k)$, Dynamic Graph Recommendation Networks starts to generates some features. There are two parts in this session. One is from item to user and the other is from user to item. From item to user, we can learn the features of the users preferences in short-term and long-term.

From user to item, we can learn the last items representation in short-term and long-term. In order to learn the long term information, the networks utilize Graph Convolution Neural Networks (GCN), Recurrent Neural Networks (GRU), and Dynamic Graph attention Mechanism (DAT). In order to learn the short-term information, the networks use attention mechanism. The last step is to use those embedding for final prediction.

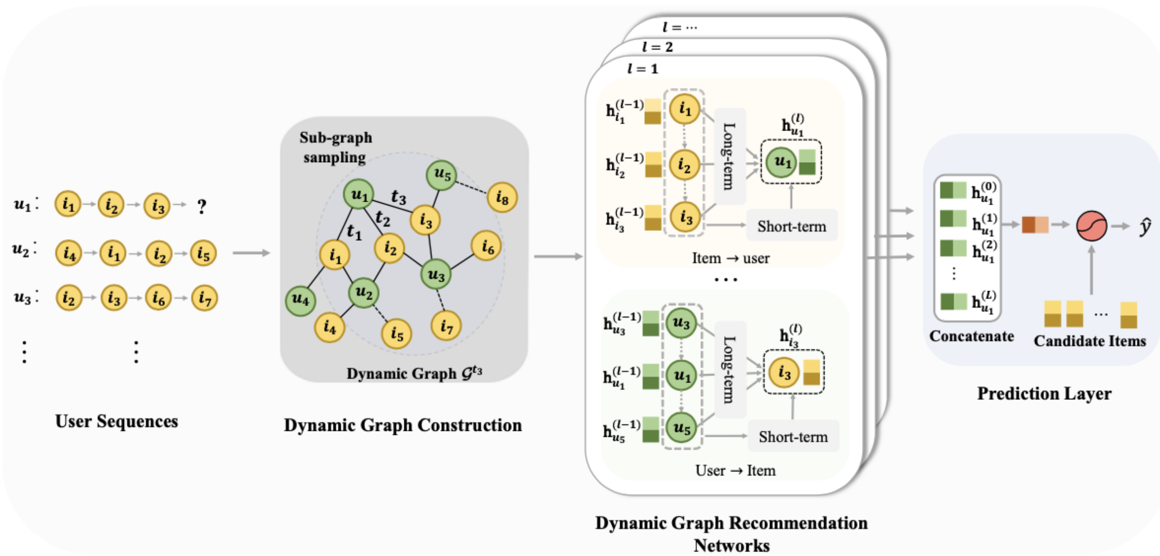

Fig. 2: Overview of DGSR framework. Take predicting the next interaction of u_1's sequence (i_1, i_2, i_3) as an example. The

Experiments

Datasets

- Amazon-CDs
- Amazon-Games
- Amazon-Beauty

Metrics

- Hit@10
- NDCG@K

Baseline

- BPR-MF
- FPMC
- GRU4Rec+
- Caser
- SASRec
- SR-GNN
- HGN
- TiSARec
- HyperRec

Results

The experiments explore three questions: one is the overall performance of this model compared with the others; the second one is how effective the different components in DGSR; the third problem is the hyper-parameter settings on DGSR.

The results shows that DGSR on those three different datasets achieve best performance comparing with the other baseline methods.

For the second question about the components in DGSR, The experiments conduct 11 variants of the model with combination of components in long-term and short-term. The results shows that DAT outperforms GCN and RNN representing that DAT can adequately extract the long-term information from neighbors of each nodes. And for the short-term information, ATT can help to extract useful information for short-term features.

The third part is about the hyper-parameters. There are DGRN Layer numbers, subgraph-sampling size, maximum sequence length and embedding size. The results shows that with the increasing of the DGRN layer number, the performance is better. For the Hit@10 metric, after the number of layer reaches to 2, the performance is stable. For the NDCG@10 metric, after 2 for Games, and 3 for Beauty, the performance will be degraded with the increase of the layer number. The experiments take 1 to 4 into account in the aspect of sub-graph sampling size. The results shows that with the size increasing, the performance will increase both for HIt@10 and NDCG@10. The performance is increasing when the embedding size increases. Increasing the maximum length of user sequence from 10 to 50, the performances increases.