Link Prediction in Citation Network

Class Project Report

Gao Fangshu
Hanqing Advanced Institute of Economics and Finance
Renmin University of China

ABSTRACT

This report presents seventh ranking solution for Link Prediction RUC in-class competition[1]. The goal of the competition is to predict citation links among papers, based on a citation network. In our approach we use different sets of features and XGBoost filter. The final F1 score on leaderboard is 0.96906.

1. INTRODUCTION

In Link Prediction RUC in-class competition, a citation network is given as a graph where nodes are research papers and there is an edge between two nodes if one of the two papers cite the other. From this citation graph, edges have been removed at random. The goal is to classify whether two nodes share a link or not.

This competition combines natural language processing and link prediction problems. Thus, we extract features from both to fit the need of the competition.

2. DATA

There are three datasets provided:

(1) `node_information.csv`: For each paper out of 27,770, contains: (a) unique node ID, (b) publication year (between 1993 and 2003), (c) title, (d) authors, (e) name of journal (not available for all papers), and (f) abstract. Abstracts are already in lowercase, common English stopwords have been removed, and punctuation marks have been removed except for intra-word dashes.

(2) `social_train.txt`: 585,512 labeled node pairs (1 if there is an edge between the two nodes, 0 else). One pair and label per row, as: source node ID, target node ID, and 1 or 0. The IDs match the papers in `node_information.csv` file.

(3) `social_test.txt`: For 30,000 node pairs we need to predict edges.

3. FEATURE EXTRACTION

3.1 Basic Features

Four sets of features can be extracted directly from data without considering graph structure:

[1]See `https://www.kaggle.com/c/socailcomputing`

3.1.1 *From publication year (1 feature)*

The year gap between two papers. For example, we get `year(A) - year(B)` if `year(A)` $\geqslant$ `year(B)`.

3.1.2 *From authors (1 feature)*

Number of overlapping authors between two paper.

3.1.3 *From title (2 features)*

For each node (paper) pair, their titles are tokenized to word lists. Then calculate Jaccard and Dice distance between the lists.

3.1.4 *From abstract (12 features)*

Tokenize abstract of each paper to word list, and take unigram/bigram/trigram of the list, with or without stopwords. Then for each node (paper) pair, calculate Jaccard and Dice distance between the n-grams. Thus we can extract $3 \times 2 \times 2 = 12$ features.

3.2 Network Features

Three networks can be established from datasets. Shown in Figure 1, they are network of journals (284 nodes), papers (27,770 nodes and 320,130 edges) and authors (11,813 nodes and 1,299,343 edges).

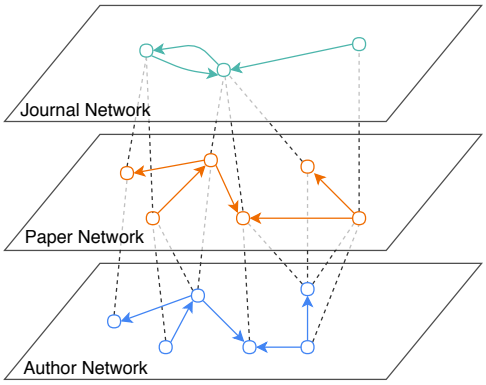

Figure 1: Three Layers of Networks

In paper network, directions of links are known from publication years, new paper always points to old paper. Note that there are 212,498 paper pairs having same publication year out of 585,512 pairs training set (Table 1), which we can not decide the directions.

Table 1: Frequency of Year Gap in Paper Network

Year Gap	Frequency
0	212,498
1	114,733
2	77,462
3	54,547
4	39,548
5	28,501
6	21,402
7	15,378
8	10,633
9	6,806
10	3,290
11	714
Total	585,512

Each paper has authors, if paper A and paper B have link from A to B (A cites B), in author network all authors of A point to authors of B.

There are 284 nodes (journals) in journal network. The graph is not clean, because most nodes do not have real journal names and have low frequency (see Table 2 in Appendix). Thus, we just extract features from paper and author network rather than journal network.

Network features can be divided into following categories:

3.2.1 *PageRank (8 features)*

Important papers or authors may be more likely to be cited. PageRank index can measure relative importance of nodes in network.

Three features are extracted from paper network: PageRank index of source nodes and target nodes in paper network, and maximum of them for each node pair.

Five features from author network: for each paper pair, take average or largest PageRank index of source nodes and target nodes in author network (one node in paper network often corresponds to many nodes in author network), and maximum of largest PageRank index.

3.2.2 *Jaccard Similarity (6 features)*

Figure 2 as an example, we want to predict whether paper A cites paper B (assume that `year(A)` $\geqslant$ `year(B)`), we know from paper network that only C_1 and C_2 cite B, and A cites only D_1 and D_2. A may be more likely to cite B if A is more similar to C_1 and C_2, or D_1 and D_2 are more similar to B.

We use Jaccard similarity to measure average or total similarity of A and Cs, or B and Ds. Note that we actually do not know the direction between A and B when `year(A)` $=$ `year(B)`, therefore maximum of (A, C) and (B, D) similarity is also extracted as a feature.

In training data, link between A and B is known, but in testing data, all nodes pairs do not have links. It can be proved that: $maxJaccard(A, C) \equiv 1$, $maxJaccard(B, D) \equiv 1$, and $meanJaccard(A, C)$, $meanJaccard(B, D)$ upward biased when A linked to B in training data. It is cheating and leads to severely overfitting. We modified the bias in our code[2].

[2]See `https://github.com/GaoFangshu/Link-Prediction/blob/master/model.py#L363`

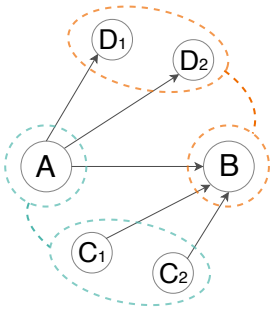

Figure 2: An Example of Paper Network

3.2.3 *Citation (8 features)*

In Figure 2, several citation features can also be added to our model:

(1) `meanAciteB`: the average number of citations the authors of B have received from authors of paper A.

(2) `maxAciteB`: the largest number of citations the authors of B have received from authors of paper A.

(3) `meanBciteA`: the average number of citations the authors of A have received from authors of paper B.

(4) `maxBciteA`: the largest number of citations the authors of A have received from authors of paper B.

(5) `maxmeancite`: maximum of `meanAciteB` and `meanBciteA`.

(6) `maxmaxcite`: maximum of `maxAciteB` and `maxBciteA`.

(7) `meanAciteB_all`: the average number of citations the authors of B have received from authors of paper A or the authors of A have received from authors of paper B.

(8) `maxAciteB_all`: the largest number of citations the authors of B have received from authors of paper A or the authors of A have received from authors of paper B.

3.2.4 *Adamic-Adar (1 feature)*

We calculate Adamic-Adar index for node pairs in paper network. It measures frequency-weighted common neighbors.

4. TRAINING AND RESULTS

Link predicting in this competition can be regarded as a binary classification problem (have link or not, given two nodes).

We use XGBoost model as classifier, but not all features in our model perform well and they are too noisy. Thus, we delete some unnecessary features by training XGBoost model repeatedly, as Figure 3 in Appendix. We get weights 1 (F score) of features after training first XGBoost model, and delete features with low weights, the remains have weights 2 in second XGBoost model. Then, we add the Jaccard similarity network features to the second model, weights of final features (weights 3) are shown in orange.

After all, the F1 of prediction on training data is 0.97280, and F1 on leaderboard is 0.96906.

5. FURTHERMORE

5.1 What We Tried

5.1.1 *Some other features*

We use Katz index measures variant of the shortest-path, the more paths there are between two papers and the shorter these paths are, the stronger the connection and the more likely the citation. However, calculating Katz index has $O(n^3)$ complexity, which is not affordable for large our graph.

5.1.2 Tuning

Exhaustive search over specified parameter values for some hyperparameters is implemented, but unfinished before deadline.

5.1.3 Ensemble

Five-fold stacking is used but does not perform well. Because we have not trained different good base models. In five-fold stacking, we combine XGBoost, Gradient Boosting, Ramdom Forest, Extra Trees Regression and Logistic Regression, and its F1 on leaderboard is 0.96360.

5.2 What Can Be Improved

(1) We just calculate Jaccard and Dice index based on N-grams for similarity. But Doc2Vec method and other word vectors (e.g. WordNet) can be used in natural language processing, which provide a deeper understanding of the abstracts.

(2) For network analysis, features from network embedding can be added. For example, Node2Vec method is convenient and often has good performance.

(3) We can extract features from combining network analysis and abstract similarity, rather than just analyze paper network itself.

(4) However, journal network may be useful, at least we can regard the importance of journal as a feature.

APPENDIX

Table 2: Frequency of Journals in Journal Network

Journal (Node) Name	Frequency
Phys.Lett.	3,575
Nucl.Phys.	3,571
Phys.Rev.	3,170
JHEP	1,957
Int.J.Mod.Phys.	938
Mod.Phys.Lett.	936
Class.Quant.Grav.	556
J.Phys.	536
J.Math.Phys.	532
Phys.Rev.Lett.	388
Phys.	377
Commun.Math.Phys.	377
Nucl.Phys.Proc.Suppl.	296
Prog.Theor.Phys.	281
Nucl.	240
Annals	207
Lett.Math.Phys.	167
Mod.	134
Fortsch.Phys.	132
Adv.Theor.Math.Phys.	126
Int.	124
Eur.Phys.J.	119
J.Geom.Phys.	91
Theor.Math.Phys.	91
Int.J.Theor.Phys.	87
Nuovo	58
Z.Phys.	57
Acta	54
Phys.Rept.	47
Prog.Theor.Phys.Suppl.	45
...	...
Leuven	1
Studies	1
Math.Res.Lett.	1
J.Phys.Stud.	1
Sov.Phys.JETP	1
Particles	1
Nonperturbative	1
lectures	1
Sov.	1
"Symmetries	1
"Black	1
Chin.Phys.	1
Matematicheskaya	1
Trends	1
Zh.Fiz.Khim.	1
Heavy	1
Theor.Math.Phys.,	1
Statistical	1
Fluctuating	1
Causality	1
Michigan	1
Investigacion	1
Nucl.Instrum.Meth.	1
Z.Naturforsch.	1
The	1
Proc	1
Rept.Prog.Phys.	1
Anales	1
Aport.	1
GROUP,	1

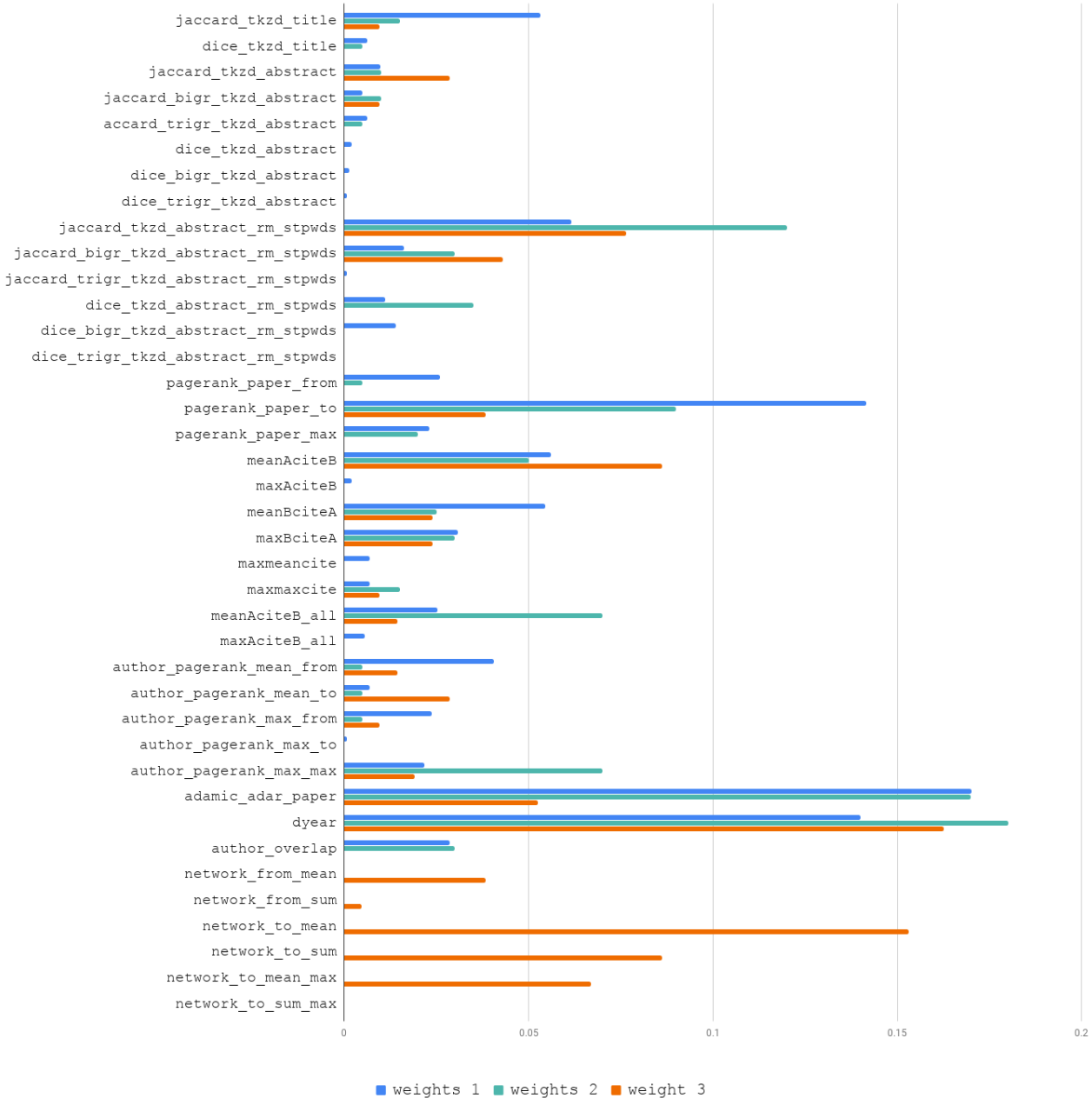

Figure 3: Feature Weights of Different XGBoost Models