GBDT and BERT: a Hybrid Solution for Recognizing Citation Intent

Zhe Wang, Rundong Shi, Shijie Li, Peng Yan
Meituan-Dianping Group
Beijing, China
{wangzhe36,shirundong,lishijie11,yanpeng04}@meituan.com

ABSTRACT

The Citation Intent Recognition task of WSDM Cup 2020 seeks a better solution to distinguish superfluous citations from genuine recognitions. In this competition, the participants are asked to develop a system that can recognize the citation intent of a given passage in a scholarly article and retrieve relevant citation targets from a given database. This paper describes the team Funny’s approach to this challenge, which win the 1st place in the Innovation Track (without using any data leaks). We formulate the retrieval task as a binary classification problem and our solution consists of the following components. Firstly, we clean the text data by some basic nlp text preprocessing methods. Secondly, we retrieve relevant citation targets from a very large dataset through a highly effective recall method. Thirdly, we extract some nlp features and train our individual models including GBDT models and neural network models. At last, we use stacking and linear blending methods for model ensemble. Our final solution achieves a public score of 0.37458 and a private score of 0.38020.

CCS CONCEPTS

• Information systems → Information retrieval.

KEYWORDS

Citation Intent Recognition, Gradient Boosting Decision Trees, BERT, Ensemble Learning

ACM Reference Format:

1 INTRODUCTION

Scientific research has become the main driving force of innovation in modern society and the academic papers have played a very important role during the entire process. In normal conditions, researchers can recognize those impactful contributions in the form of their larger number of citations. However, more superfluous and coercive citations have appeared in recent years for the purpose of promotions or winning competitions of research funding. How to apply the web search and data mining technologies to distinguish superfluous citations from genuine recognitions becomes a very meaningful and challenging problem. In this competition, participants are required to develop a system that can recognize the citation intent of a given passage in a scholarly article and retrieve relevant citation targets from a given database.

The competition takes the Mean Average Precision@3 (MAP@3) as the evaluation metric. The required output is a list of maximum 3 papers for each item ordered by predicted citation probability for a specific paper. The higher the actually cited paper appears on the list the higher the score is. The evaluation formula is as followed:

\[
MAP@3 = \frac{1}{U} \sum_{u=1}^{U} \min(3, n) \sum_{k=1}^{\min(3, n)} P(k)
\]

where \( U \) is the number of items, \( n \) is the number of predictions and \( P(k) \) is the score of the particular position (range from 1 to 3). Specifically, if the prediction in the first position is correct, the score is 1, if the prediction in the second position is correct, the score is 1/2, if the prediction in the third position is correct, the score is 1/3.

This paper describes the team Funny’s approach to this challenge, which win the 1st place in the Innovation Track (without using any data leaks). The rest of the paper is organized as follows. We first provide a brief data analysis and describe our text data clean method in Section 2. Section 3 is about candidates recall. Our ranking solution is in Section 4 which includes feature engineering, models building and blending. Experiment results are illustrated in Section 5. Finally, we make a conclusion in Section 6. The overall framework of our approach is shown in Figure 1.

2 DATASET

The competition asks participants to retrieve the top 3 relative papers to a given passage in a scholarly article. The data set provided by the sponsor consists of three files: the candidate paper set (candidate.csv), the training set (train_release.csv) and the public/private test set (validation.csv/test.csv). The candidate paper dataset contains about 800K papers. Some of them have been matched to the given passage in the training set and the task is to find the other cited papers for those passages in the public/private test set. The field descriptions of the candidate set and the training set are shown in Table 1 and Table 2 respectively. The public/private test set are in the same format as the training set, but the ‘paper_id’ column is empty for prediction.
2.1 Data Analysis

Exploratory data analysis (EDA) is very important to any data mining task. We first do EDA on the given data sets. The training data set consists of 62,976 description-paper pairs, and the candidate data set contains information of 838,939 papers. We find that some important information in the candidate set are missing (shown in Table 3) and the sentence length distributions for both abstract and description vary a lot (shown in Fig. 2). Additionally, we also find that some records in the training set with different description_id have the same description_text. We remove these duplicated records before feature extraction and model training.

Table 1: Field descriptions of the candidate set

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>paper_id</td>
<td>paper’s id</td>
</tr>
<tr>
<td>title</td>
<td>paper’s title</td>
</tr>
<tr>
<td>abstract</td>
<td>paper’s abstract</td>
</tr>
<tr>
<td>journal</td>
<td>in which journal the paper was published</td>
</tr>
<tr>
<td>year</td>
<td>paper’s publication year</td>
</tr>
<tr>
<td>keyword</td>
<td>paper’s keywords</td>
</tr>
</tbody>
</table>

Table 2: Field description of the training set

<table>
<thead>
<tr>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>description_id</td>
<td>description id</td>
</tr>
<tr>
<td>paper_id</td>
<td>cited paper’s id</td>
</tr>
<tr>
<td>description_text</td>
<td>description content</td>
</tr>
<tr>
<td></td>
<td>(the original index is replaced by ”[<strong>##</strong>]”).</td>
</tr>
</tbody>
</table>

Table 3: Examples of the papers with missing information

<table>
<thead>
<tr>
<th>paper_id</th>
<th>title</th>
<th>keywords</th>
<th>abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td>55a38b7f2401-aa93797ce61</td>
<td>A method of computing...</td>
<td>NaN</td>
<td>NO_CONTENT</td>
</tr>
<tr>
<td>55a56b48240-12c2a39230188</td>
<td>Studies on the...</td>
<td>NaN</td>
<td>NO_CONTENT</td>
</tr>
</tbody>
</table>

2.2 Data Preprocessing

For most of natural language processing (NLP) tasks, text preprocessing is helpful for improving the final performance. In the competition, we also process the texts based on the results of EDA. Firstly, we transform the text data to lower case and map some special characters like α and alpha to the same word. Then we remove those useless punctuations especially empty space and keep the stem of the words based on part-of-speech analysis. Many of candidate papers provided no abstract and keywords, therefore we concat the paper’s title, abstract and keywords as a new text. On the other hand, we also extract the text of the given passage which is near to the reference notation, since this part may be most relevant to the cited paper. For convenience of describing the following models and features, we first introduce some notations.

- query_key: the text of the given passage which nearby the reference notation
- query_all: the whole part of the original given passage
- paper_title: the title of the candidate paper
- paper_abstract: the abstract of the candidate paper

Figure 1: The overall framework of our approach
GBDT and BERT: a Hybrid Solution for Recognizing Citation Intent

3 RECALL

In order to retrieve relevant citation targets from a large given database, we first recall top 50 papers for each item according to their similarity scores. We have compared different methods for computing similarity scores such as tf-idf vector based cosine similarity and bm25 score between \{query_key, query_all\} and \{paper_title, paper_abstract, paper_keyword, paper_content\}. In practice, bm25 score between query_all and paper_content performs the best. We try to recall a larger number of articles, but it takes more time and improves little. At last, we set the recall number to 50.

4 RANK

4.1 GBDT Models

Firstly, we formulate the task as a binary classification problem. After obtaining the top 50 recall candidates, we regard the papers truly cited as positive instances while the other are treated as negative ones thus the proportion between positive and negative samples is about 1:50. The classifier will output a probability, representing the citation probability to a specific paper. We choose GBDT [5] as our classification model which is a popular machine learning algorithm, and has quite a few effective implementations such as XGBoost [2] and LightGBM [6]. The major advantage of GBDT is its high efficiency and the superior ability to find nonlinear interactions automatically. We use LightGBM as our final model with parameters num_leaves=32, feature_fraction=0.7, bagging_fraction=0.7, learning_rate=0.03. Our model converges at around 500 iterations with early_stopping_round=50. The features we used for training GBDT models are consist of three major parts: basic features, distance features and pairwise features.

4.1.1 Basic Features. We generate counter features for \{query_key, query_all, paper_title, paper_abstract, paper_keyword, paper_content\} such as count of n-gram, number of unique n-gram, count and ratio of digits. Further more, we also generate intersect counter features between \{query_key, query_all\} and \{paper_title, paper_abstract, paper_keyword, paper_content\}. In addition, for those intersect n-gram, we record their positions, and compute the following statistics as features such as minimum, median, maximum, mean and standard deviation.

4.1.2 Distance Features. We generate distance features between \{query_key, query_all\} and \{paper_title, paper_abstract, paper_keyword, paper_content\}. Firstly, we calculate the Jaccard coefficient and Dice distance between the n-gram of the previous pairs. Then, we compute the cosine similarity, euclidean distance and manhattan based on their tf-idf vectors. Finally, we also calculate their bm25 scores.

4.1.3 Pairwise Features. The key point to predict the cited paper is to find the paper’s advantages against others. Therefore, for each item, we sort candidate papers by their model predictions and create pairwise features by computing the difference between the top-k paper’ average model predictions and the current one [7].

4.2 Neural Network Models

4.2.1 Wide&Deep. At the beginning, we try some conventional text matching models and find that they are not good at this problem. Through the analysis of badcases, we think that these models may not be able to capture the information of text similarity because the sample size is too small. The effect of word vector pre-training using the text of the candidate papers improves MAP@3 score but the score is still not satisfactory. Finally, we use the Wide&Deep [3] structure to supplement the features that neural networks do not capture. We use the ESM [1] as the deep part of the model and use the top 30 features of GBDT feature as the wide part.

4.2.2 BERT. In the paper, we also use BERT [4] as our single model. BERT is a language representation model designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers. BERT-base model contains an encoder with 12 transformer blocks, 12 self-attention heads, and the hidden size of 768 (the model architecture is shown in Fig. 3). It takes an input of a sequence of no more than 512 tokens and outputs the representation of the sequence. The sequence has two segments and the first token of every sequence is always a special classification token \([CLS]\). We fine-tuned the Google’s pre-trained BERT models. For this task, BERT takes the final hidden state of the first token \([CLS]\) as the representation of the whole sequence. A simple fully connected network is added to predict the matching score of query and document. To adapt BERT to the task, we consider several factors:

- The first one is the preprocessing of long text since the maximum sequence length of BERT is 512. We use WordPiece embeddings with a 30,000 token vocabulary and use tail-only method of truncate text.
- The second one is the overfitting problem. On the one hand, we choose the appropriate optimization method. We use adam with lr=5e-5, β1=0.9, β2=0.999, batch_size=8, train steps of 100,000 and warm-up steps of 10,000. The dropout probability of BERT is kept at 0.1 and the dropout probability of added fully connected network is kept at 0.2. On the other hand, we use 5-fold early stopping strategy to prevent overfitting.
- The third one is optimization target. We use pairwise loss \(P_{ij} = \frac{1}{1+e^{-(reg_1*reg_2)}}\) as loss function.
Table 4: Performance of different models

<table>
<thead>
<tr>
<th>Model</th>
<th>Public MAP@3</th>
<th>Private MAP@3</th>
</tr>
</thead>
<tbody>
<tr>
<td>GBDT</td>
<td>0.3163</td>
<td>-</td>
</tr>
<tr>
<td>Wide&amp;Deep</td>
<td>0.3164</td>
<td>-</td>
</tr>
<tr>
<td>Bert</td>
<td>0.3381</td>
<td>0.3394</td>
</tr>
<tr>
<td>Linear Blending</td>
<td>0.3746</td>
<td>0.3802</td>
</tr>
</tbody>
</table>

- The fourth one is model diversity. We trained two type of BERT models: one is query_all as query and the other is query_key as query.

4.3 Ensemble Modeling

Ensemble modeling combines the decisions from multiple models to improve the overall performance. Simple ensemble techniques include max voting, averaging and weighted average, while some advanced techniques are more effective such as bagging, boosting and stacking. We use stacking method in this context. Stacking uses predictions from multiple models (e.g. decision tree, k-NN and SVM) to build a new model which is used for making predictions on the cross-validation set and the test set. Traditional stacking methods only use predictions as input features, and the model usually converge very fast. In this competition, we not only use previous models’ predictions but also add many important original features especially distance features and pairwise features. In addition, for stacking inputs, we elaborately select suitable model predictions from our complete model candidates. At last, we use linear blending to ensemble the stacking models and some BERT models to generate our final result.

5 EXPERIMENTS

We evaluate our method on the Citation Intent Recognition task of WSDM Cup 2020 data set, and the detailed experiment results are shown in Table 4.

6 CONCLUSION

In this paper, we summarize our solution to the Citation Intent Recognition task of WSDM Cup 2020. Our experiments show that well designed features and BERT are especially effective for information retrieval problems which won us, the Funny team, the 1st place in the Innovation Track (without using any data leaks). Besides, ensemble modeling, a commonly used effective technique, also helps us in improving our model performance. The source code is available at https://github.com/wsdm-Teamfunny/wsdm2020-solution and we will verify the validity of this method in other areas in the future.

7 ACKNOWLEDGEMENT

In the finals period (last two days), some teams found data leaks of the test set and used it to improve private leaderboard score a lot. And finally, to encourage more effective solutions, the organizers decide to set an Innovation Track and recalculate the final ranks by performance scores without using any data leaks. We express our sincere appreciation to the organizers of WSDM Cup 2020 for recalculating the ranks, which leads us a winner at last. And we are glad to have a chance to share our solution in public. We also thank Guolin Ke, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu from Microsoft and Qi Meng from Peking University for providing LightGBM, such a high efficient implementation of GBDT and Google researchers for BERT.

REFERENCES