An Empirical Two-Stage Framework on Citation Intent Recognition

The 2nd place entry for Citation Intent Recognition Competition at WSDM Cup 2020

Kuei-Chun Huang
National Taiwan University
Taipei, Taiwan
r08922010@ntu.edu.tw

Chi-Yu Yang
National Taiwan University
Taipei, Taiwan
r08922a15@ntu.edu.tw

ABSTRACT

Known as "coercive citations", journal editors are seen to force authors to cite marginally relevant articles in particular journals to boost their journal impact factors, so are paper reviewers to solely increase their citation counts or h-index. In this contest, the contestant is 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. In this paper, we describe our winning approach for finding most related papers. In particular, we will explore the application of Bidirectional Encoder Representations from Transformers (BERT) [6] and Sent2Vec [7] for embedding sentence pairs, which can not only make ranking model more robust but also recall more various papers so as to improve recalling accuracy. The proposed solution of our team *SimpleBaseline* achieved weighted accuracy score of 0.41712 in the private leaderboard, and was selected as second place submission.[1]

KEYWORDS

Citation Intent Recognition, WSDM Cup

ACM Reference Format:
Kuei-Chun Huang and Chi-Yu Yang. 2020. An Empirical Two-Stage Framework on Citation Intent Recognition : The 2nd place entry for Citation Intent Recognition Competition at WSDM Cup 2020. In *Proceedings of WSDM conference (WSDM'20).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1]The source code is available at https://github.com/steven95421/WSDM_SimpleBaseline

WSDM'20, February 2020, Houston, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The WSDM 2020 Citation Intent Recognition contest presents research descriptions requiring the three most relative papers from a large dataset which contains roughly 800K papers. To address this challenge, we introduce five different recalling approaches including BM25, IDF, Sent2Vec, BlueBERT [8] and Keywords, which are used to recall papers from different aspects. Also, for each recalled paper we prepare over 200 features for LightGBM [5] model to train and predict the probability of correct citation. Our approach achieved 0.41712 and 0.41441 weighted accuracy in the private and public leader boards, respectively.

The rest of the paper is organized as follows: Section 2 introduces the dataset of the competition. In Section 3, we describe our solution which contains the model details. In Section 4, we show the experiment results of our model. Finally, we conclude our analysis of the contest, as well as some additional discussions of the future directions in Section 5.

2 DATASET

There are a training dataset, a testing dataset, and a candidate-paper pool for this contest. Candidate-paper pool contains 838,939 papers with the following attributes:

- `paper_id`: paper's ID (missing: 1)
- `title`: paper's title (missing: 0)
- `abstract`: paper's abstract (missing: 69,252)
- `journal`: on which journal the paper was published (missing: 35,249)
- `year`: publication time (missing: 234,820)
- `keywords`: paper's keywords (missing: 777,694)

Training dataset consists of 62,976 pieces of description texts with the following columns:

- `description_id`: ID of description
- `paper_id`: ID of paper of correct citation
- `description_text`: text of descriptions. The original index is replaced by "[**##**]"

Testing dataset consists of 34,428 pieces of description texts with the same columns as training dataset except "paper_id".

2.1 Preprocess Text

For every text in training data, testing data, and candidate papers, we first lemmatized every word, next removed stop words and special characters, then lower the text. Additionally, we extracted key sentences the same way as description in contest website. But for key sentences less than 7 words we replaced them with corresponding preprocessed description texts directly.

3 PROPOSED APPROACH

Due to the size of candidate-paper pool, generating features for all papers directly will be totally infeasible. To handle this challenge, we made a plan with two stages including recall and ranking. In recall stage several unsupervised methods were built to reduce the scope of candidates, then we prepared a ton of features for a binary-classifying model to rank the candidate papers selected in the recalling stage.

3.1 Recalling Stage

For BM25 and IDF, we concatenated preprocessed `title`, `abstract`, and `keywords` and used all papers in candidate pool as corpus. Similarily, we used preprocessed key sentences to find the top N most matching papers. As for Sent2Vec, we should skip the lemmatizing step during preprocessing data since lemmatized words might become Out-of-Vocabulary. Using previously preprocessed data directly would cause a terrible 2% drop in recall@3. However, this issue only affected BlueBERT a little, we thus employed the same input as BM25 to BlueBERT for convenience.

3.1.1 BM25

In information retrieval, Okapi BM25 (BM stands for Best Matching) is a ranking function used by search engines to rank matching documents according to their relevance to a given search query, representing state-of-the-art TF-IDF-like retrieval functions used in document retrieval. BM25 has two tunable parameters b and k_1; the former controls how much effect field-length normalization should have, while the latter controls how quickly an increase in term frequency results in term-frequency saturation. In this task, we sticked with the default values ($b = 0.75$, $k_1 = 1.5$) since which performed best.

3.1.2 IDF

The Term Frequency-Inverse Document Frequency (TF-IDF) algorithm is the most common computation used in text processing and information retrieval applications. This is a statistical quantity used to measure the importance of a word with respect to a document corpus. However, we found that in this task term frequency was not so important that we could improve recalling accuracy by around 1% if we only use Inverse Document Frequency (IDF). In addition, we also dicovered that words with lower IDF value were too common to be representative of a document. Therefore, we modified IDFs as below:

$$IDF_i = \begin{cases} 0 & \text{if } IDF_i < 5 \\ IDF_i & \text{otherwise} \end{cases}$$

3.1.3 Sent2Vec

A sentence-embedding model with simple but efficient unsupervised training objective. The algorithm outperforms the state-of-the-art unsupervised models on most benchmark tasks, and on many tasks even beats supervised models, highlighting the robustness of the produced sentence embeddings. Conceptually, the model can be interpreted as a natural extension of the word-contexts from C-BOW to a larger sentence context, with the sentence words being specifically optimized towards additive combination over the sentence, by means of the unsupervised objective function. In this contest we could easily see that the description texts were from biology kind of papers. Therefore, we used "BioSentVec_PubMed_MIMICIII-bigram_d700.bin" as underlying word-embedding model for Sent2Vec.

3.1.4 BlueBERT

The BERT model architecture is based on a multilayer bidirectional Transformer. Instead of the traditional left-to-right language modeling objective, BERT is trained on two tasks: predicting randomly masked tokens and predicting whether two sentences follow each other. BERT model gets a lot of state of the arts in many tasks. In this task we used BlueBERT, which was trained on biology papers, to add a few recalls in a totally different aspect. Although there is an more famous BERT model also trained on biology papers called BioBERT, we found that BlueBERT outperforms BioBERT in this task.

3.1.5 Keywords

From 3.1.1 to 3.1.4, we discovered that traditional IR methods outperformed these deep learning ones. According to the previous observation, keywords usually have two characteristics: long length and high IDF value. Therefore, we purposed a method to extract keyword from the sentences: top-k longest words with IDF value higher than certain threshold [2]. Then, we recalled the papers by calculating the number of overlapping keywords generated by the above procedure. Despite the fact that the recalling rate was not quite well, this recalling method was still essential for our performance. With only 2 recalled papers added, we improved our overall recalling accuracy by more than 1%, which showed how distinguished the papers recalled by this approach were.

[2] We set $k = 10$ and $threshold = 13$

3.1.6 Combination

Finally after several times of tuning, we recalled 87 papers for each description (50 from BM25, 10 from IDF, 20 from Sent2Vec, 5 from BlueBERT, and 2 from Keywords), which achieved recalling accuracy 0.60436.

Table 1. Recalling accuracy

method	recall@3	recall@10	recall@30	recall@87
BM25	0.310	0.395	0.475	0.557
IDF	0.169	0.248	0.330	0.429
Sent2Vec	0.178	0.269	0.360	0.462
BlueBERT	0.076	0.125	0.186	0.257
Keywords	0.066	0.110	0.148	0.167

3.2 Feature Engineering

In this task we were asked to find the most matching papers for each description, therefore, the "distance" between description and paper must be quite important for model to learn. Whatsoever, we still employed two different directions of features for each recalled pair, and there were 217 features in total. The First was the attributes of description and paper, while the second was the distances between description and paper. [3] The following subsections will show the details of above two kinds of features.

3.2.1 Attributes

Adding attribute features is quite simple. For description, we computed the length of description text and key sentence. As for paper, we just prepared two boolean variables whether the paper has abstract and whether it has keywords.

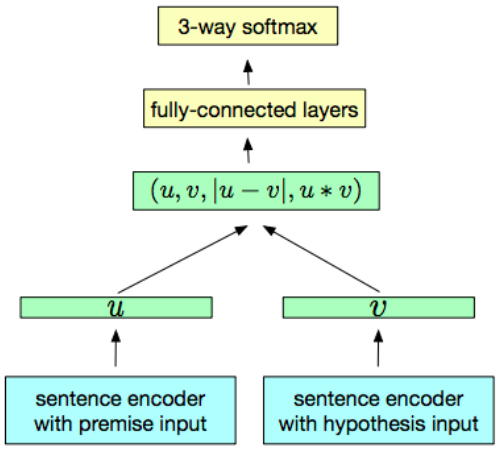

Figure 1: **Generic NLI training scheme.**

3.2.2 Distances

For the raw text we computed several distances including Jaccard similarity, Levenshtein distance, Jaro distance, jaro-winkler distance. Besides, we also calculated the BM25 score and the number and rate of overlapping common words, keywords, upper words [4], and pseudo keywords [5]. As For each kind of embedding we computed distances including cosine similarity, Manhattan distance, and Euclidean distance between description and paper. The followings will introduce the embeddings we used in this task:

- **IDF**: The same embedding as section 3.1.2
- **Word2Vec** [9]: Word2Vec is a three-layer neural network, In which the first is the input layer and the last layers are the output layer. The middle layer builds a latent representation so the input words transformed into the output vector representation. We used average of word embeddings to be sentence embedding.
- **FastText** [3]: An approach based on the skipgram model, where each word is represented as a bag of character n-grams. A vector representation is associated to each character n-gram; words being represented as the sum of these representations. This method is fast, allowing to train models on large corpora quickly and to compute word representations for words that did not appear in the training data. We also used average of word embeddings to be sentence embedding.
- **SIF** [2]: SIF computes sentence embeddings as a weighted average of word vectors. We took "BioWordVec_PubMed_MIMICIII_d200.bin" as underlying word embeddings. First, compute all the frequencies of all the words of corpus. Then, given a set of pre-trained word embeddings, compute the weighted average above for each sentence. Finally, Use SVD to remove the first component off of these averages and get fresh sentence embeddings. Considering its simplicity, SIF embeddings perform very well. In fact, this method can even outperform state-of-the-art deep learning techniques such as InferSent on semantic textual similarity tasks.
- **Sent2Vec**: The same embedding as section 3.1.3
- **BlueBERT**: The same embedding as section 3.1.4
- **SciBERT** [1]: SciBERT is trained on papers from the corpus of semanticscholar.org, using the full text of the papers, not just abstracts. Corpus size is 1.14M papers, 3.1B tokens. It has its own vocabulary (scivocab) that's built to best match the training corpus, resulting in state-of-the-art performance on a wide range of scientific domain nlp tasks.

[3] Each description has description text and key sentence, and each paper has title and abstract. Therefore, there are four distances in all.

[4] Acronyms must be representative of a sentence.

[5] Generated by the same way as section 3.1.5 with several thresholds.

- **Fine-tuning SciBERT**: We used InferSent [4], as illustrated in Figure 1 and replaced sentence encoders with SciBERT. In particular for this embedding, we only generated embeddings of descriptions' description text and papers' title since it took us too much time to produce embeddings before the closing time of the competition.

3.3 Ranking Stage

Gradient boosting is a powerful algorithm especially for binary-classifying problem. Recently, LightGBM has gained increased popularity and attention due to their advantages of fast processing speed and high prediction performance. We utilized LightGBM to build 10 models with different feature sets split by stratified 10-fold cross-validation. The hyperparameter values we have to train the models are as follows:

- num_leaves: 64
- reg_alpha: 1
- reg_lambda: 0.1
- objective: 'binary'
- max_depth: -1
- learning_rate: 0.1
- min_child_samples: 5
- n_estimators: 5000
- subsample: 0.8
- colsample_bytree: 0.8

4 EXPERIMENTS

We tried 4 approaches to ensemble ten LightGBM models: [6]

- **Average**: Calculate average of logits from ten models.
- **Vote with linear weights**: $weight_i = 9 - i$, for top i-th paper.
- **Vote with reciprocal weights**: $weight_i = 1/i$, for top i-th paper.
- **Vote with KR weights**: Inspired by online game "Kart Rider", which gives scores to 8 riders by ranking of a game with $rank2weight = \{1:10, 2:8, 3:6, 4:5, 5:4, 6:3, 7:2, 8:1\}$

Table 2. Performance of various ensembling

method	weighted accuracy
Average	0.41187
Vote with linear weights	0.41389
Vote with reciprocal weights	0.41367
Vote with KR weights	**0.41441**

Table 2 lists the results of various approaches described previously. Voting with KR weights got the best performance of 0.41441, which also outperformed other methods no matter how we modified features during development period and was 1st place on the public leaderboard. Hence, We only employed this strategy on the private leaderboard and achieved 2nd place on the final leaderboard with score 0.41712.

[6]For voting method we employed top 8 papers from each model, which was obtained from several times of trials.

5 CONCLUSIONS

In this paper, we have introduced an empirical framework for the Citation Intent Recognition Competition of the WSDM Cup 2020. Our team *SimpleBaseline* was ranked the second place on the final leaderboard. In our solution, we first conducted various recalling methods to reduce the scope of candidate papers, then we prepared over 200 features for classifying model to learn. After that, we trained 10 LightGBM models using stratified 10-fold cross-validation. Finally, we ensembled these 10 models by voting strategy with weights inspired by "Kart Rider". While obtaining promising performance on the whole, our model still cannot handle some bad cases. We will leave these challenges for future work. For example, since there are still tons of Out-of-Vocabulary words in three datasets, we can pre-train or continue training the BERT models and the two biological word-embedding models mentioned in this paper using news documents.

REFERENCES

[1] Sanjeev Arora, Yingyu Liang, Tengyu MIz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: Pretrained Language Model for Scientific Text. *arXiv preprint arXiv:1903.10676* (2019).

[2] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. *International Conference on Learning Representations* (2017).

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).

[4] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 670–680. https://www.aclweb.org/anthology/D17-1070

[5] Thomas Finley Taifeng Wang Wei Chen Weidong Ma Qiwei Ye Tie-Yan Liu Guolin Ke, Qi Meng. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. *Advances in Neural Information Processing Systems 30* (2017), 3149–3157.

[6] Kenton Lee Jacob Devlin, Ming-Wei Chang and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. *Conference of the North American Chapter of the Association for Computational Linguistics* (2018).

[8] Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets. *Proceedings of the 2019 Workshop on Biomedical Natural Language Processing* (2019).

[9] Greg Corrado Jeffrey Dean Tomas Mikolov, Kai Chen. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781* (2013).