IR&TM-NJUST@CLSciSumm 20

Heng Zhang, Lifan Liu, Ruping Wang, Shaohu Hu, Shutian Ma, Chengzhi Zhang*

Department of Information Management, Nanjing University of Science and Technology, Nanjing, China, 210094
zh_heng@njust.edu.cn, liulf@njust.edu.cn, 2935843497@qq.com,
191226105@qq.com, mashutian0608@hotmail.com, zhangcz@njust.edu.cn

Abstract

This paper mainly introduces our methods for Task 1A and Task 1B of CL-SciSumm 2020. Task 1A is to identify reference text in reference paper. Traditional machine learning models and MLP model are used. We evaluate the performances of these models and submit the final results from the optimal model. Compared with previous work, we optimize the ratio of positive to negative examples after data sampling. In order to construct features for classification, we calculate similarities between reference text and candidate sentences based on sentence vectors. Accordingly, nine similarities are used, of which eight are chosen from what we used in CL-SciSumm 2019 and a new sentence similarity based on fastText is added. Task 1B is to classify the facets of reference text. Unlike the methods used in CL-SciSumm 2019, we construct inputs of models based on word vectors and add deep learning models for classification this year.

1 Introduction

The rapid growth of papers has provided scholars with various knowledge and methods, which can offer references for development or innovation of the research. But it makes difficult for researchers to get brief summaries quickly from such massive amount of papers (Radev et al., 2002). Automatic summarization can solve this problem. Researchers express their views on reference paper through citation text. So, citation text can be used to generate summary of paper (Cohan & Goharian, 2018; Qazvinian & Radev, 2008). However, as a result of researchers' different views (citation), the quality of the summary is not guaranteed and the summary cannot fully restore the original information of paper. Therefore, CL-SciSumm proposes to generate summary by the original text corresponding to citation. CL-SciSumm is the first medium-scale shared task on scientific document summarization, with over 500 annotated documents [1]. This competition is organized annually from 2016, and we can view details about CL-SciSumm2020 at the website: https://ornlcda.github.io/SDProc/sharedtasks.html #clscisumm. The introduction of CL-SciSumm2020 is as follows:

Given: *A topic consisting of a Reference Paper (RP) and Citing Papers (CPs) that all contain citations to the RP. In each CP, the text spans (i.e., citances) have been identified that pertain to a particular citation to the RP.*

Task 1A: *For each citance, identify the spans of text (cited text spans) in the RP that most accurately reflect the citance. These are of the granularity of a sentence fragment, a full sentence, or several consecutive sentences (no more than 5).*

Task 1B: *For each cited text span, identify what facet of the paper it belongs to, from a predefined set of facets.*

Task 2 (optional bonus task): *Finally, generate a structured summary of the RP from the cited text spans of the RP. The length of the summary should not exceed 250 words.*

In Figure 1, The blue text span in the citing paper shows the citation text, and the green text span in the reference paper shows the reference text which most accurately reflects the citance.

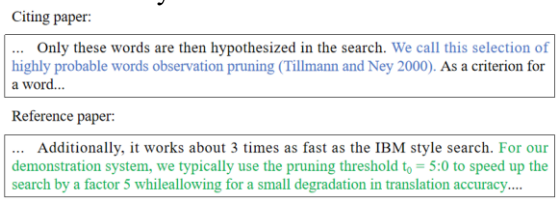

Figure 1: Citation text in citing paper and reference text in reference paper

* Corresponding Author.

[1] https://github.com/WING-NUS/scisumm-corpus/

Proceedings of the First Workshop on Scholarly Document Processing, pages 288–296

https://doi.org/10.18653/v1/P17

Our team has participated in the CL-SciSumm competition in 2017 (Ma et al., 2017), 2018 (Ma, et al., 2018) and 2019 (Ma et al., 2019). For Task 1A, a similarity-based negative sampling strategy is applied to construct the training set. Nine similarity features and sentence vectors are used to represent citation text and candidate sentences. Then we employ traditional machine learning methods and build MLP model to identify the reference text in reference papers. For Task 1B, sentence vectors are generated based on word frequency and word vector. Traditional machine learning models and deep learning models are built to identify the facets. As for Task 2, cosine similarity is calculated between reference sentences and the original abstract based on their sentence vectors. Then sentences are selected to construct summary according to their similarities, and length of the summary does not exceed 250 words.

Compared with previous work, we make changes in following steps. In Task 1A, we optimize ratio of positive to negative examples after negative sampling. The structure and parameters of MLP model are adjusted to get better results. For Task 1B, we first try to use word vector to construct inputs of models. And the result has been improved about 10% at accuracy score.

2 Related works

2.1 Identification of the citation text spans

As for the related work of Task 1A, most previous teams solved it by using classification models, and they constructed different features as input of models. Some researchers used three types of classification features, namely similarity-based features, rule-based features and location-based features (Jaidka et al., 2017). Ma et al. (2017) extracted several features at the words level from the citation text spans in the training set to calculate the corresponding similarities, such as IDF similarity, Jaccard similarity, Dice similarity, Word2Vec similarity and so on.

In recent years, machine learning models are mostly used for the identification of citation text spans. Mei and Zhai (2008) highlighted the importance of citance, and they proposed a method to generate the abstract of the cited document by extracting the most influential sentences in the document. The machine learning models mainly include classification models and ranking models. Yeh et al. (2017) used classification models, such as SVM (Support Vector Machines), DT (decision trees), KNN (K-Nearest Neighbors) and so on in the identification of citances. Their method performed well with competitive results when it was evaluated using the CL-SciSumm 2016 datasets. In ranking models, sentences were sorted based on the integration of multiple features. Lu et al. (2016) constructed word-level (e.g. TF-IDF similarity and Jaccard similarity) and topic-level features (based on LDA model) separately and used the learning-to-rank algorithm to identify cited text spans. Their results showed that Jaccard similarity achieved better F measures, and the performance of topic similarity features varies slightly among different number of topics. Additionally, Moraes et al. (2016) investigated cosine similarity with multiple incremental modifications and SVMs with a tree kernel. They calculated the similarity not only between reference and citance sentences, but also between the reference spans and the citance sentences.

In summary, the current research about identification of citation text spans mainly includes feature construction and model selection. Most of the researches attempt to construct a huge feature system for model training and learning. As for model selection, most of the works are based on traditional machine learning models or sorting algorithms.

2.2 Identification of the facets of reference text

Task 1B is to identify the facets of reference text. It provides 5 facets in this task. Most teams in previous CL-SciSumm competitions used rule-based methods, because the amounts of different facets of reference text are imbalanced (Ma et al., 2018). In the learning process of the classification algorithms, the result tends to focus on the facets with most samples. This problem will have a huge impact on model training (He & Garcia, 2009). He et al. (2008) reviewed researches about learning from imbalanced data, then they highlighted that the opportunities and challenges to solve this problem would be a new research field in the future research. Ma, et al. (2018) combined the NN algorithm with the SMOTE algorithm to make training data and extend the penalty factor in the processing of imbalanced datasets, and NN algorithm behaved best on testing data.

There are plenty of researches about identifying the facets of reference text, rule-based methods and statistical-based methods are widely used. Wang et al. (2012) proposed an orderly clue phrase matching method and got 62% accuracy and 42% recall. Sándor et al. (2006) presented two natural language processing systems to help researchers rapidly accessing relevant knowledge in text. Agarwal et al. (2011) used two statistical machine learning models, SVM and NB, to classify the facets of reference. And they found that the classification result of SVM was better. Aggarwal and Sharma (2016) determined the facets based on the location of the cited text spans. Li et al. (2019) used the Word2Vec and the CNN model to calculate the sentence similarity, and further apply CNN to classify the facets of reference texts. They indicated that the features of high frequency word and subtitle are important in the identification of facets.

In summary, in the researches about classification of facets, the approaches applied in this task mainly include rule-based methods and statistical-based methods. However, because of the limited experimental dataset and the imbalance in the number of samples in different facets, these two methods are difficult to learn the relevant features of the facets more accurately and efficiently.

3 Methodology

Before introducing the methodology of each task, we define some concepts to avoid ambiguity in the following description.

Table 1: Concepts and their definitions

Concept	Definition
Citation text	It is "Citance" in introduction of Task 1A, and it consists of one or several sentences from citing paper. See blue highlighted span in Figure 1.
Reference text	It is "cited text spans" in introduction of Task 1B, and it consists of one or several sentences from reference paper. See green highlighted span in Figure 1.
Facets	It is the type of reference text, there is a predefined set of facets: "Method_Citation", "Result_Citation", "Aim_Citation", "Implication_Citation", "Hypothesis_Citation".
Candidate sentences	Citation text and candidate sentences as a pair of input to models. And candidate sentences contain reference text as positive samples and sentences selected from reference paper as negative samples.

3.1 Task 1A based on negative sampling

In Task 1A, we are given citation text to find the corresponding sentences in the reference paper. This task can be regarded as a binary classification task. For a citation text, it is need to identify the classification labels of all sentences in the reference paper. There are two classification labels: "1" or "0". If "1", it means that the sentence belongs to the correct reference text. If "0", it means that the sentence is not. Figure 2 shows our research framework of Task 1A. Firstly, preprocessing is conducted for the data extracted from data set. Secondly, training data is constructed by negative sampling. Then, nine similarities are calculated between citation text and candidate sentences, which are used as features to construct input of traditional machine learning models. Additionally, MLP model is built based on sentence vector. Finally, these models are evaluated with Precision (P), Recall (R), and F_1-value (F_1).

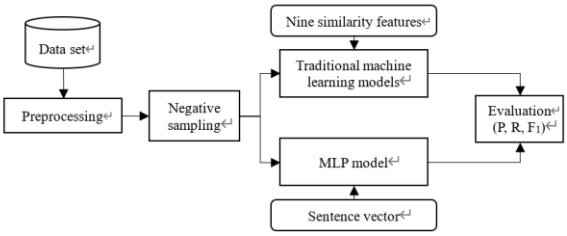

Figure 2: Framework of Task 1A

Negative sampling: 753 pairs of citation text and reference text are extracted from annotation in "Training-Set-2018", and they are used as positive samples (label "1"). Citation text and other arbitrary sentences in reference papers can be regarded as negative samples (label "0"), but the number of negative samples is too huge. In order to balance positive and negative samples, negative sampling based on sentence vector similarity is performed. We calculate the average of all word vectors in the sentence and obtain a new vector to represent the sentence. Then, cosine similarities are calculated between the citation text and all sentences in reference paper (apart from the reference text annotated). Next, sentences are chosen from the highest, lowest, and middle similarity levels to form negative samples. Through comparative experiments, the ratio of the number of positive to negative samples is finally determined as 1:6 (two sentences with the highest

similarity, two sentences with the lowest similarity, and two sentences with medium similarity as negative samples).

Using traditional machine learning models to identify reference text: The first idea is to use traditional machine learning methods to solve Task 1A. We calculate multiple similarities between citation text and candidate sentences as features. It is worth noting that candidate sentences contain reference text and 6 negative samples, citation text and reference text are regarded as a whole respectively to calculate their sentence vectors. Nine similarity indicators are selected and they are showed in Table 2. Then several machine learning models are trained for classification. These models contain Support Vector Machine (SVM) (Cortes and Vapnik, 1995), Naive Bayesian (NB) (McCallum et al., 1998), K-Nearest Neighbor (KNN) (Altman, 1992), Decision Tree (DT) (Quinlan, 1987), Random Forest (RF) (Ho, 1995) and ensemble learning tool (Xgboost[2]).

Table 2: Nine similarities as features

Similarity	Description
Jaccard similarity	Segment setence1 and setence2 into set of words, denoted as s_1 and s_2 respectively, and calculate the division of the intersection and union between two sets. Its formulation is as follows: $J(s_1,s_2)=\frac{len(s_1 \cap s_2)}{len(s_1)+len(s_2)-len(s_1 \cap$
Dice similarity	Segment setence1 and setence2 into sets of words(s_1, s_2). Its formulation is as follows: $\frac{2 * intersection(s_1,s_2)}{length(s_1)+length(s_2)}$
Word Overlap	Segment setence1 and setence2 into sets of words, and calculate the number of overlaps between them.
Bigram Overlap	Segment setence1 and setence2 into sets of bigrams, and calculate the number of overlaps between them.
Longest Common Subsequence	Denote setence1 and setence2 as two sets of sequences with words as basic unit, find the longest subsequence (not necessarily consecutive in original sequences) common of them.
Longest Common Substring	Denote setence1 and setence2 as two sets of strings with words as basic units, and find the longest string(s) that is a substring(s) (required to occupy consecutive positions within the original strings) of them.
Levenshtein distance	Calculate the average of Levenshtein distance (the minimum number of single character edits required to change one to the other) for all the words between setence1 and setence2.
Word2Vec similarity	Represent words as low-dimensional and dense distributed representation by Word2Vec algorithm and calculate the average of the similarity between words from two sentences via cosine value.
fastText similarity [3]	Represent words as low-dimensional and dense distributed representation by fastText algorithm and calculate the average of the similarity between words from two sentences via cosine value.

Using MLP model to identify reference text: The second idea is to use deep learning models. Word2Vec(Mikolov et al., 2013) and fastText are used to train word vectors. And we calculate the average of all word vectors in sentence to get sentence vectors. Vector of citation text and vector of candidate sentence are concatenated as input of models. We build MLP model and adjust hidden layers and parameters for optimization.

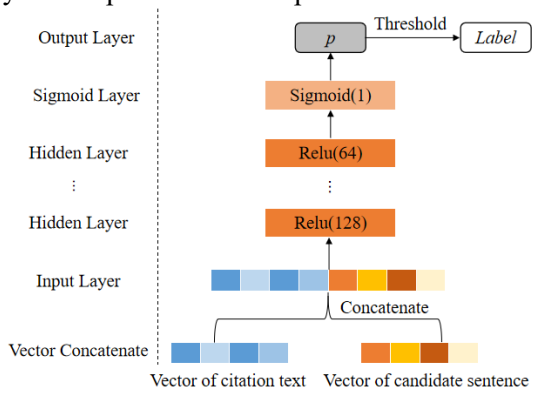

Figure 3: Framework of MLP model in Task 1A

The framework of MLP model is shown in Figure 3. The input of the model is concatenated sentence vector from citation text and reference text. Concatenated sentence vector passes through two hidden layers, and then passes through the sigmoid layer. We get the probability of two labels through the output layer and set a threshold to determine which label the candidate sentence belongs to. It should be noted that the activation

[2] https://github.com/dmlc/xgboost

[3] https://github.com/facebookresearch/fastText

function of the hidden layer is Relu, and the number of neural nodes is 128 and 64 respectively. These parameters are finally determined based on comparative experiments.

3.2 Task 1B based on sentence vector and word embedding

In Task 1B, it is a multi-label classification task. There are five labels (facets): "Method_Citation", "Result_Citation", "Aim_Citation", "Implication_Citation", "Hypothesis_Citation". The research framework of Task 1B is shown in Figure 4. Firstly, 753 pairs of citation text and reference text is extracted from data set. Secondly, training set and test set are split from the extracted data by sampling. Then, sentence vectors are generated from word frequency and word vector based on which traditional machine learning models are used to classify the facets. In addition, the word embedding matrix is used as input, and deep learning models are also applied in Task 1B. In order to test the effects of different models, accuracy score is used.

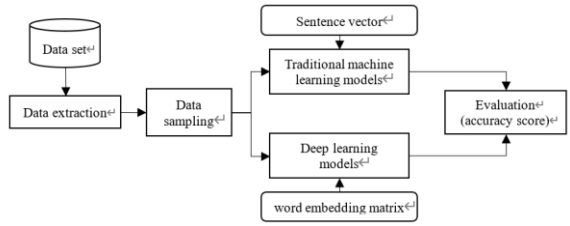

Figure 4: Framework of Task 1B

Data sampling: The number of samples in five facets varies greatly (see Figure 5). Training set and test set should not be divided from all the samples directly. In order to balance all kinds of samples in training set and test set, we randomly select 80% of samples from each label to form training set, and the remaining 20% of the samples are used as test set.

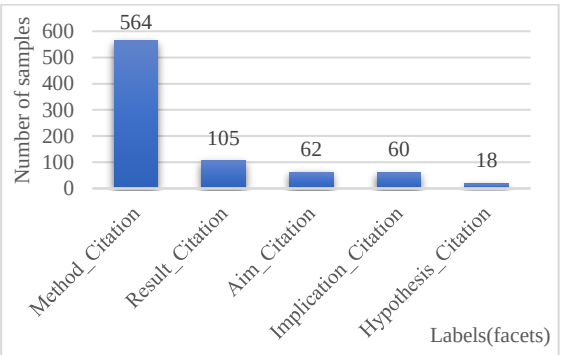

Figure 5: Number of samples in each label

Using traditional machine learning models to identify the facets based on sentence vector: As illustrated in the framework, traditional machine learning models are employed in Task 1B based on the input of sentence vectors. By the way, sentence vectors are generated from word frequency and word vector separately. In the first way, nouns, verbs, adverbs, adjectives are selected after part-of-speech tagging. Then, sentence vectors are generated by One-hot or TF (Term Frequency) based on the selected words. In the second way, fastText and BERT[4] are used to train word vector. And we calculate the average of all word vectors in the sentence to generate the sentence vector. After that, traditional machine learning models introduced in Task 1A are used for the multi-label classification. Besides, we add another ensemble learning tool LightGBM[5]. During testing, if the model cannot assign a label to a sample, we will set the sample's label to "Method_Citation".".

Using deep learning models to identify the facets based on word embedding: We also build deep learning models for the multi-label classification in Task 1B. In this scheme, word embedding matrix is used as input. Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) and Recurrent Neural Network (RNN) (Rumelhart et al., 1986) are applied in the feature selection layer separately. They convert the word embedding matrix into a 128-dimensional vector. Then the vector passes through a hidden layer, and we get the probabilities that the sample belongs to five labels. When the probability is greater than 0.5, we assign the corresponding label to the sample. If the sample fails to obtain a label, we set its label to "Method_Citation".

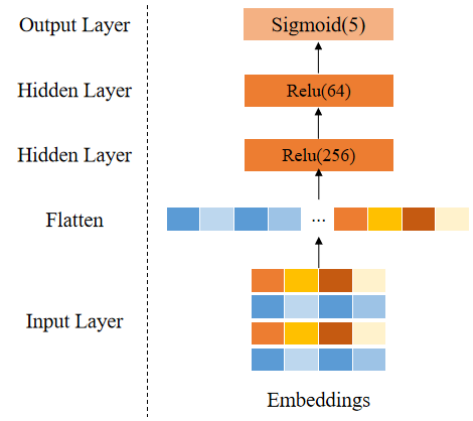

Figure 6: Framework of MLP model in Task 1B

[4] https://github.com/google-research/bert

[5] https://github.com/microsoft/LightGBM

In Figure 5, We build an MLP model for Task 1B. The word embedding matrix is flatted into a vector, and the vector pass through two hidden layers. Finally, the model outputs the probabilities that the sample belongs to five labels.

3.3 Task 2 based on sentence similarity

In Task 2, we select sentences from reference text by calculating cosine similarity between the sentence and the original abstract to generate abstract. The steps are as follows:

a. Word vector is trained by fastText.

b. Sentence vectors of reference sentences (identified in Task 1A) and the original abstract are generated by calculating the average of vectors of all words in the sentence.

c. Calculate cosine similarity between reference sentences and the original abstract based on their sentence vectors.

d. Select sentences according to their similarities to generate summary, and length of the summary does not exceed 250 words.

4 Experiments and results analysis

In this section, we report the results of different models in Task 1A and Task 1B.

4.1 Experimental result of Task 1A

For task 1A, we use nine similarities as features and applied traditional machine learning models to identify reference text. MLP model is also employed based on the input of sentence vector. In this section, we report and analysis the results of these models.

Results of traditional machine learning models: Input of sentence vector is generated based on nine similarities. And five classification models in Scikit-learn[6]: Random Forest, Decision Tree, SVM, NB, KNN are applied. In addition, ensemble learning model by Xgboost is employed. Precision, Recall, and F_1-value are used to evaluate their performance. The results of 5-fold cross validation are shown in Table 3.

Table 3: Evaluation results of models

Model	P	R	F_1
Xgboost	0.5124	0.5449	**0.5280**
Random Forest	0.6732	0.4087	0.5084
Decision Tree	0.4680	0.4442	0.4550
SVM	0.6415	0.3168	0.4230
NB	0.2626	0.9430	0.4106
KNN	0.4957	0.3345	0.3987

From Table 3, we can see that ensemble learning method by Xgboost achieves the optimal F_1-value.

Results of MLP model: Word vectors are trained through two tools: Word2Vec and fastText. The training corpus consists of two parts: (1) Full-text of reference papers and citing papers from "Training-Set-2018". (2) Full-text of reference papers from "ScisummNet-2019". The vector dimension is set to 200. Through comparative experiments, we finally determined the optimal parameter settings under these two kinds of word vector, as shown in Table 4.

Table 4: Parameters of MLP models

Model	MLP_FT	MLP_FT
Word vector	fastText	Word2Vec
Optimizer	adam	RMSprop
Loss	binary_cross entropy	mse
Epoch	20	20
Hidden layer	Rule (128) Rule (64)	Rule (128) Rule (64)
Threshold	0.577	0.602

The evaluation results of these two models are shown in Table 5.

Table 5: Evaluation results of MLP models

Model	P	R	F_1
MLP_FT	0.6486	0.6316	**0.6400**
MLP_W2Vs	0.6428	0.5684	0.6034

As surfaced in Table 5, the results based on fastText is better than Word2Vec. F_1-value of the best result is 0.64. Compared with the results of machine learning models, MLP works better.

But when we use the trained models to identify the sentences in reference papers for citation text, the models output far more than 5 sentences. In order to ensure the effect of the final test, we develop a sentence filtering strategy in reference papers:

a. We pick out nouns in citation text and sentences of reference papers.

b. In reference paper, sentences with the same noun as citation text are filtered out.

c. We use trained models to identify the filtered sentences. Because we find that 609 of the 753 pairs of citation text and reference text have the same nouns.

d. When the final test, if there is no sentence with

[6] https://scikit-learn.org/stable/index.html

the same noun as citation text in the reference paper, we will test all sentences in the reference paper.

4.2 Experimental results of Task 1B

For Task 1B, sentence vector and word embedding matrix are used as input. Then traditional machine learning models and deep learning models are applied for the multi-label classification. Now, we report and analysis the results of these models.

Accuracy score of traditional machine learning models based on one-hot: Sentence vectors are generated by one-hot in two ways. (1) Nouns, verbs, adverbs and adjectives are only selected in citation text. (2) Nouns, verbs, adverbs and adjectives are selected in both citation text and reference text. Many machine learning models in Scikit-learn and ensemble learning models by Xgboost and LightGBM are applied for classification. Accuracy score is used to evaluate these models. Random Forest and two ensemble models work better, and their accuracy scores are demonstrated in Table 6.

Table 6: Evaluation results of models based on One-hot

Model	citation text	citation text and reference text
Random Forest	0.7580	**0.8025**
Xgboost	0.7134	0.6688
LightGBM	0.7707	0.7962

From Table 6, when sentence vectors are generated by One-hot based on citation text and reference text, Random Forest works better and its accuracy score is 0.8025.

Accuracy score of traditional machine learning models based on TF (Term Frequency): We also use TF to generate vectors in two ways: citation text, citation text and reference text. Evaluation results of Random Forest, Xgboost and LightGBM are shown in Table 7.

Table 7: Evaluation results of models based on TF

Model	citation text	citation text and reference text
Random Forest	0.7580	**0.7962**
Xgboost	0.7134	0.7580
LightGBM	0.6624	**0.7962**

As suggested in Table 7, when sentence vectors are generated by TF based on citation text and reference text, Random Forest and LightGBM achieve higher accuracy score.

Accuracy_score of traditional machine learning models based on fastText: Sentence vectors are generated based on fastText word vector. Sentence vector of citation text is recorded as $v_1 = (x_1, x_2 \ldots x_n)$, and sentence vector of reference text is recorded as $v_2 = (y_1, y_2 \ldots y_n)$. We also calculate $|v_1\text{-}v_2| = (|x_1\text{-}y_1|, |x_2\text{-}y_2| \ldots |x_n\text{-}y_n|)$ and $v_1 * v_2 = (x_1 * y_1, x_2 * y_2 \ldots x_n * y_n)$. We make four combinations of v_1 and v_2:

a. $(v_1, v_2) = (x_1, x_2 \ldots x_n, y_1, y_2 \ldots y_n)$

b. $(v_1, v_2, |v_1\text{-}v_2|) = (x_1, x_2 \ldots x_n, y_1, y_2 \ldots y_n, |x_1\text{-}y_1|, |x_2\text{-}y_2| \ldots |x_n\text{-}y_n|)$

c. $(v_1, v_2, v_1 * v_2) = (x_1, x_2 \ldots x_n, y_1, y_2 \ldots y_n, x_1 * y_1, x_2 * y_2 \ldots x_n * y_n)$

d. $(v_1, v_2, |v_1\text{-}v_2|, v_1 * v_2) = (x_1, x_2 \ldots x_n, y_1, y_2 \ldots y_n, |x_1\text{-}y_1|, |x_2\text{-}y_2| \ldots |x_n\text{-}y_n|, x_1 * y_1, x_2 * y_2 \ldots x_n * y_n)$

In each combination, vectors are concatenated as input of different models. Evaluation results of Random Forest, Xgboost and LightGBM are shown in Figure 7.

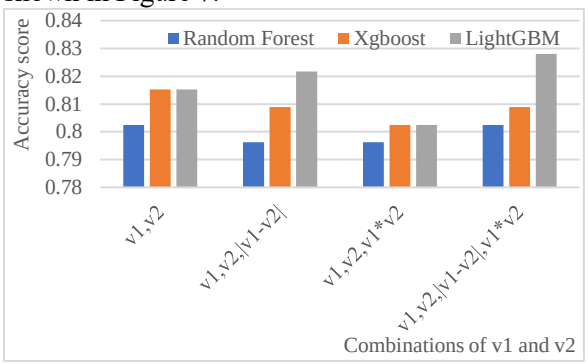

Figure 7: Evaluation results of models based on fastText

As shown in Figure 7, under different conditions, LightGBM performs better than the other two models. When v_1, v_2, $|v_1\text{-}v_2|$ and $v_1 * v_2$ are concatenated as input, LightGBM reaches the highest accuracy score (0.8280).

Accuracy score of traditional machine learning models based on BERT: We train word vector by BERT and calculate sentence vectors. Evaluation results of three models are shown in Figure 8.

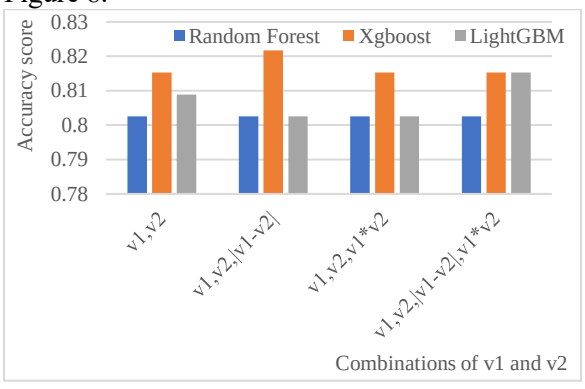

Figure 8: Evaluation results of models based on BERT

As illustrated in Figure 8, under different conditions, Xgboost performs better than the other two models. When v_1, v_2, and $|v_1-v_2|$ are concatenated as input, Xgboost get the highest accuracy score (0.8217). But its performance is slightly worse than LightGBM with fastText word vectors (see Figure 7).

Accuracy score of deep learning models based on word embedding: Word vectors trained by fastText and BERT are used to construct word embedding matrix of citation text and reference text. Then three deep learning models: LSTM, RNN and MLP are applied with the input of word embedding matrix. Accuracy score of the three models are shown in Figure 9.

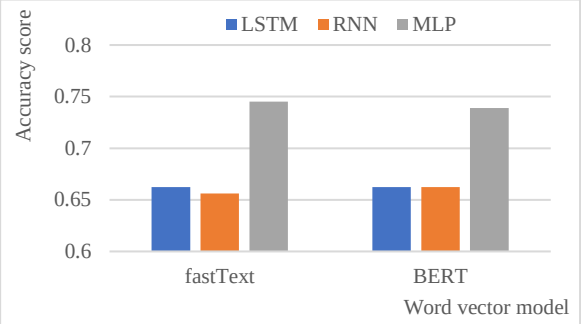

Figure 9: Evaluation results of deep learning models

From Figure 9, we can see that MLP performs best among the three models. But its accuracy score is lower than the previous results of LightGBM and Xgboost (see Figure 7 and Figure 8).

5 Conclusion and future work

In Task 1A, training data and test data are constructed by negative sampling. And the ratio of positive to negative examples has been optimized. Next, we use deep learning model (MLP) with the input of sentence vectors and traditional machine learning models based on nine similarity features to identify the reference text. The effect of MLP is proved to be better than that of traditional machine learning models. As for Task 1B, we calculate different combinations of sentence vectors as input. Traditional machine learning models and deep learning models have been evaluated on classifying the facets of reference text. In this process, the effect of using pre-training model (BERT) to obtain word vector is worse than that of using fastText to train word vector based on training set. And traditional machine models (LightGBM and Xgboost) work better than deep learning models.

Generally, word vectors can reflect more semantic information compared to traditional machine learning features. We create a suitable number of training data by negative sampling in Task 1A, so deep learning model (MLP) works better. While in Task 1B, insufficient training data makes deep learning models inferior to traditional machine learning models.

In future work, we can optimize training set through Data Augmentation Technology and apply other deep learning models for Task 1A. As for Task 1B, its recognition result is affected by the imbalance of data. We will try to expand the training data for the facets with smaller data scale from other data sources, such as structured abstract.

Acknowledgements

This work is supported by National Natural Science Foundation of China (Grant No. 72074113).

Reference

Agarwal, N. K., Xu, Y. (Calvin), & Poo, D. C. C. (2011). A context-based investigation into source use by information seekers. Journal of the American Society for Information Science and Technology, 62(6), 1087-1104. https://doi.org/10.1002/asi.21513

Aggarwal, P., & Sharma, R. (2016). Lexical and Syntactic cues to identify Reference Scope of Citance. Proceedings of the Joint Workshop on Bibliometric-Enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL), 103-112.

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician, 46(3), 175-185.

Cohan, A., & Goharian, N. (2018). Scientific document summarization via citation contextualization and scientific discourse. International Journal on Digital Libraries, 19(2-3), 287-303.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273-297. https://doi.org/10.1007/BF00994018

He, H., & Garcia, E. A. (2009). Learning from Imbalanced Data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263-1284. https://doi.org/10.1109/TKDE.2008.239

He, H., Yang Bai, Garcia, E. A., & Shutao Li. (2008). ADASYN: Adaptive synthetic sampling approach

for imbalanced learning. 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 1322‑1328. https://doi.org/10.1109/IJCNN.2008.4633969

Ho, T. K. (1995). Random decision forests. Proceedings of 3rd International Conference on Document Analysis and Recognition, 1, 278‑282.

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Comput., 9(8), 1735‑1780. https://doi.org/10.1162/neco.1997.9.8.1735

Jaidka, K., Chandrasekaran, M. K., Jain, D., & Kan, M.-Y. (2017). The CL-SciSumm Shared Task 2017: Results and Key Insights. BIRNDL@ SIGIR (2).

Li, L., Zhu, Y., Xie, Y., Huang, Z., Liu, W., Li, X., & Liu, Y. (2019). CIST@ CLSciSumm-19: Automatic Scientific Paper Summarization with Citances and Facets. BIRNDL@ SIGIR, 196‑207.

Lu, K., Mao, J., Li, G., & Xu, J. (2016). Recognizing reference spans and classifying their discourse facets. Proceedings of the Joint Workshop on Bibliometric-Enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL), 139‑145.

Ma, S., Xu, J., Wang, J., & Zhang, C. (2017). NJUST @ CLSciSumm-17. Proceedings of the Joint Workshop on Bibliometric-Enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2017).

Ma, S., Xu, J., & Zhang, C. (2018). Automatic identification of cited text spans: A multi-classifier approach over imbalanced dataset. Scientometrics, 116(2), 1303‑1330. https://doi.org/10.1007/s11192-018-2754-2

Ma, S., Zhang, H., Xu, J., & Zhang, C. (2018). NJUST @ CLSciSumm-18. BIRNDL@ SIGIR.

Ma, S., Zhang, H., Xu, T., Xu, J., Hu, S., & Zhang, C. (2019). IR&TM-NJUST@ CLSciSumm-19. BIRNDL@ SIGIR, 181‑195.

McCallum, A., Nigam, K., & others. (1998). A comparison of event models for naive bayes text classification. Proceedings of the AAAI-98 Workshop on Learning for Text Categorization, 752(1), 41‑48.

Mei, Q., & Zhai, C. (2008). Generating impact-based summaries for scientific literature. Proceedings of ACL-08: HLT, 816‑824.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. ArXiv:1301.3781 [Cs]. http://arxiv.org/abs/1301.3781

Moraes, L., Baki, S., Verma, R., & Lee, D. (2016). University of Houston at CL-SciSumm 2016: SVMs with tree kernels and Sentence Similarity. Proceedings of the Joint Workshop on Bibliometric-Enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL), 113‑121.

Qazvinian, V., & Radev, D. R. (2008). Scientific paper summarization using citation summary networks. ArXiv Preprint ArXiv:0807.1560.

Quinlan, J. R. (1987). Simplifying decision trees. International Journal of Man-Machine Studies, 27(3), 221‑234.

Radev, D. R., Hovy, E., & McKeown, K. (2002). Introduction to the special issue on summarization. Computational Linguistics, 28(4), 399‑408.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533‑536. https://doi.org/10.1038/323533a0

Sándor, Á., Kaplan, A., & Rondeau, G. (2006). Discourse and citation analysis with concept-matching. International Symposium: Discourse and Document (ISDD), 15‑16.

Wang, W., Villavicencio, P., & Watanabe, T. (2012). Analysis of reference relationships among research papers, based on citation context. International Journal on Artificial Intelligence Tools, 21(02), 1240004. https://doi.org/10.1142/S0218213012400040

Yeh, J.-Y., Hsu, T.-Y., Tsai, C.-J., & Cheng, P.-C. (2017). Reference Scope Identification for Citances by Classification with Text Similarity Measures. Proceedings of the 6th International Conference on Software and Computer Applications, 87‑91. https://doi.org/10.1145/3056662.3056692