SemEval-2018 Task 7 Subtask 1: Semantic Relation Classification in Scientific Papers

Hui-Qiang Jiang Da-Wei Lee
Peking University
{1801210840,1701210963}@pku.edu.cn

April 9, 2019

Abstract

First, we test the performance with some naive ideas to transform the problem into a trainable form. Then sifting out the unfeasible one and bad performance one. Doing some research on previous competitors' paper. Then we decide to follow the LightRel model. We looking for the most important factors that affect the performance the most. Finally, focusing on them and aim to gain more improvement on the Macro F1 score.

We achieve a 56.3% F1 score on Task 1.1. And it beats the best performance of the LightRel 46.4%.

1 Introduction

Semantic relation extraction and classification is a typical task in Nature Language Processing. These relations are successfully used in various NLP applications, such as word sense disambiguation, query expansion, article summarization, question answer or machine translation. A task of semantic relation classification (SRC) is defined as predicting a semantic relationship between two tagged entities in sentences.

However, previous semantic relation classification models rely heavily on high-level lexical and syntactic features obtained from NLP tools, such as part-of-speech (POS) tagger, dependency parser, and named entity recognizer (NER). The classification model relying on such features suffer from the propagation of implicit error od the tools and they are computationally expensive.

Most approaches focus on modeling a single semantic relation and consist in deciding whether a given relation holds between a pair of words (x, y) or not. Another research direction consists of dealing with multiple semantic relations at a time and can be defined as deciding which semantic.

And there exist many types of domain-specific semantic relations between words (entities) expert open-domain semantic relations. In scientific paper abstracts, there are lots of relations between words like usage, topic, result, model, part-whole, etc. Our task is to classify the semantic relation on some scientific paper dataset (SemEval-2018).

In this paper, we propose a way fusion end-to-end method and traditional feature engineer. It is philosophically similar to LightRel [Renslow and Neumann(2018)] which showed the idea is fruitful for SemEval-2018 Task 7. And we use oversampling to solve the problem of dataset imbalance on subtask1.1.

2 Related Work

We spent much time working on how to transform the problem into a trainable form. We found that it is also very hard for a human (ourselves) to classify this task. (And we have drawn on the experience of the other paper.)

2.1 Naive Idea on Relation

We first doing some research on semantic relation typology. There are six major relation types of this competition, USAGE, RESULT, MODEL, PART_WHOLE, TOPIC, and COMPARISON. And each of them may have a REVERSE relation (except COMPARISON).

The idea is putting connection words between two entities. And calculate the probability of whether it matches which relation type the most.

There are the combinations we've made:

- USAGE
  - used by
  - used for
  - applied to
  - performed on
- RESULT
  - affects
  - problem
  - yields
- MODEL
  - of and observed
  - associated to
- PART_WHOLE
  - composed of
  - extracted from
  - found in
- TOPIC
  - presents
  - of
- COMPARISON
  - compared to

2.2 Improved Relation

We try to use the words between two entities as the basis of predicting relation. And we found that LightRel has done a similar thing.

LightRel let all the sentences with the same length (i.e. same dimension feature). But the sentence in the training and test set won't always be the same. Thus they have padded some dummy words into the sentence to fill the empty space.

We think this approach is much more reasonable than the previous one. Because the relation of any two words can easily tell by the connecting words or sentence.

2.3 Feature Engineering

We have also observed some of the training samples. And conclude some possible pattern such that if the first word is end with "'s", then it possibly has maybe a sort of belonging relation between them.

In LightRel, they have done many features, they called "word-shape feature", as well. But by our experiment, we found that. Using these features will only lower the performance. So we only use enable this as comparison purpose.

- any character is capitalized
- a comma is present
- the first character is capitalized and the word is the first in the relation
- the first character is lower-case
- there is an underscore present (representing a multi-word entity)
- if quotes are present in the token

2.4 Embedding

We found that the most significant improvement of the performance is related to embedding. There are two subjects: What corpus we choose to calculate the embedding. And what tools to form the embedding. We have made several tests.

2.4.1 Corpus

We selecting the candidate corpus on the Citation Network Dataset [1] [Tang et al.(2008)Tang, Zhang, Yao, Li, Zhang, and Su]. This dataset provides the scientific paper of recent years. LightRel chooses DBLP v5 combined with ACM v9. And extract out only the "Abstract" part of the paper using the regular expression.

We have done some combination of selection like testing by using DBLP v5 only or using DBLP v10 combined with ACM v9 etc.

2.4.2 Embedding Model

There are several tools that we have tried. Such as word2vec [Mikolov et al.(2013)Mikolov, Sutskever, Chen, Corrado, and Dean] by Google (also used in LightRel), fastText by Facebook and BERT by Google.

We build the embedding with 300 dimension vector and skip the words with appearance less than 5 times. We use the mean of all other embeddings to deal with the out-of-vocabulary problem (i.e. the ¡UNK¿ token).

There are some tests among these three model. The usage of the word2vec and the fastText is very similar. The BERT, because there are too many parameters, need to finetune to fit the problem, but we haven't found a good solution, thus the performance is not as good as the previous tools. Finally, we found that fastText perform better than others.

3 Model and Framework

The first attempt is continuing using the model used by LightRel, the logistic regression model offered by LibLinear. But to form the required format that fit LibLinear is quite handy because it needs to be file format with a kind of sparse matrix representation. That become complicated when we need to do k-fold cross-validation on the training set.

Then we tried to use Scikit Learn library. Because the API of Scikit Learn are similar so we can switch from multiple models and test the performance easily. We have tried LinearSVC, LogisticRegression and Decision TreeClassifier. And we found the LogisticRegression has still had the best performance.

In the test of LightGBM, the performance is not quite ideal. In the first case, we think that because the dimension of features is too large, using a tree-based algorithm may not be a good idea.

Finally, we have tried the simple version TextCNN model, the performance is not too bad but still lower than the statistics machine learning model. So we have deprecated it later on.

4 Experiment

In the previous section, we mentioned many experiments on the different corpus, embedding tools and model. We will list the performance of each attempt and highlight the best performance on both tasks.

Because the TOPIC class in subtask 1.1 is unbalanced. So we can see the phenomenon that subtask 1.2 will overall be higher than subtask 1.1.

[1] https://aminer.org/citation

4.1 Different Corpus and Embedding Model

Word embedding almost the important thing in NLP task. So we take some word embedding method and train dataSet to tune the effect of word embedding. 1

First of all, the train dataSet of competition is so small that the effect of training word embedding is bad. So we import some outer dataSet to optimization the effect of word embedding. The domain of our task is about scientific papers. So, we load dataset on Citation Network Dataset. We test dblp v5, dblp v10, and acm v9, we find dblp v10 have best f1 score both in trainSet and testSet.

We also do some work on different word embedding methods, like word2vec, Bert, fastText. FatsText do the best job in our experiment. Bert doesn't have a good effect on our task. We think it may be caused by the embedding size of the pre-train model.

Task		Subtask 1.1 Test		Subtask 1.1 Training	
Corpus	Embedding Model	Macro-F1	Micro-F1	Macro-F1	Micro-F1
Pre-trained DBLP v5	word2vec	44.61	-	**50.79**	-
ACM v9 + DBLP v10	word2vec	47.24	-	49.45	-
DBLP v5	word2vec	46.27	-	50.08	-
DBLP v10	word2vec	47.28	-	50.28	-
DBLP v5	fastText	49.12	-	49.30	-
ACM v9 + DBLP v10	fastText	50.21	-	50.73	-
DBLP v10	fastText	**50.75**	-	49.72	-
ACM v9 + DBLP v10	BERT	26.02	-	32.82	-

Table 1: Comparison between different corpus using LibLinear logistic regression model on subtask 1.1 data (in %)

4.2 Different Feature

We also do some work on feature engineering. 2 We test the effect of different '[PAD]' position. Clustering is also one way to improve our model. We add both one-hot and word embedding of middle words between two entities to feature lists. And we also do some artificial features like have including '_' in middle words between two entities. We do some experiment which one by one add features to model.

4.3 Different Model

4.4 Imbalance Data

5 Conclusion and Future Work

References

[Mikolov et al.(2013)Mikolov, Sutskever, Chen, Corrado, and Dean] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of

Task		Subtask 1.1 Test		Subtask 1.1 Training	
Model	Feature	Macro-F1	Micro-F1	Macro-F1	Micro-F1
LibLinear Logistic Regression	embedding only	**52.01**	-	49.97	-
LibLinear Logistic Regression	+ one-hot	51.43	-	49.25	-
LibLinear Logistic Regression	+ shape	50.61	-	49.67	-
LibLinear Logistic Regression	+ cluster	51.55	-	49.31	-
LibLinear Logistic Regression	+ e1, e2	49.93	-	49.97	-
Scikit Learn Logistic Regression	embedding only	50.37	-	52.12	-
Scikit Learn Logistic Regression	+ one-hot	51.10	-	52.37	-
Scikit Learn Logistic Regression	+ shape	51.29	-	51.81	-
Scikit Learn Logistic Regression	+ cluster	51.90	-	51.97	-
Scikit Learn Logistic Regression	+ e1, e2	50.92	-	**52.99**	-

Table 2: Comparison between differen feature based on LibLinear and Scikit Learn logistic regression model (in %)

Task	Subtask 1.1		Subtask 1.2	
Model Name	Macro-F1	Micro-F1	Macro-F1	Micro-F1
LibLinear Logistic Regression	52.01	-	69.15	**81.41**
Scikit Learn Logistic Regression	**52.77**	-	75.06	76.04
Scikit Learn Linear SVM	51.45	-	70.29	75.56
Scikit Learn Decision Tree	36.65	-	-	-
TextCNN	51.20	65.50	**77.35**	80.53

Table 3: Comparison between differen machine learning model using fastText embedding model on DBLP v10 corpus (in %)

words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

[Renslow and Neumann(2018)] Tyler Renslow and Günter Neumann. 2018. Lightrel semeval-2018 task 7: Lightweight and fast relation classification. *arXiv preprint arXiv:1804.08426*.

[Tang et al.(2008)Tang, Zhang, Yao, Li, Zhang, and Su] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: Extraction and mining of academic social networks. In *KDD'08*, pages 990–998.