Lincoln Samelson
12/19/2018
Ling 506
Final Project

Automatic Discourse Relation Classifier

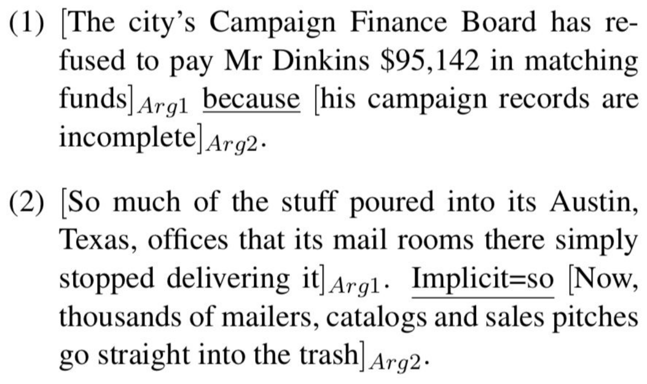
0 Introduction

Discourse relation classification is a crucial component for developing an automatic discourse parser. Discourse parsing is an important sub field in computational linguistics. Without the development of discourse parsing, the future of a natural language system that is able to have fluid dialogue with a human is slim to none. If discourse parsing was not developed, one cannot build a system that is able to relate any phrases in a discourse. The system would be completely incapable of having a fluid dialogue. It would be akin to an elderly person not remembering the previous discourse. But with the development of an accurate discourse relation classifier, it will ultimately help lead to a more sophisticated and better performing automatic discourse parser.

1 Problem Definition

This project will focus on classifying discourse relations. The end product is to design a classification model that will be able to take in new input and automatically classify the relation. Specially, classifying explicit, implicit, altLex, entRel, and noRel relations. For instance, in figure (1) below, shows an example of 2 clauses joined by a connective, relating the two clauses. The connective being because. This is an explicit relation since there is a connective that is explicitly present, *because*, connecting the two clauses. Figure (2) exemplifies an implicit relation, also between two clauses. There is no explicit connective present connecting the two clauses. It’s simply implied that all of the mail will be thrown into the trash, because of the overflowing mail rooms. The task is to build a system that is able to automatically classify these relations between clauses.

As for the input of the system, we will be using the penn discourse treebank corpus to train and eventually test the classifier. The penn discourse treebank includes a large annotated discourse corpus along with useful meta data for each pair of clauses and includes the pair of clauses, their relation, connective (if applicable), syntactic tree structure, etc. Though the corpus is large and very useful for this project, it will need to be cleaned and formatted to properly fit the python learning model (sklearn). The output will be a confusion matrix for each relation class which will help evaluate how well the finished model performed.



(1) [The city’s Campaign Finance Board has refused to pay Mr Dinkins $95,142 in matching funds]$_{Arg1}$ because [his campaign records are incomplete]$_{Arg2}$.

(2) [So much of the stuff poured into its Austin, Texas, offices that its mail rooms there simply stopped delivering it]$_{Arg1}$. Implicit=so [Now, thousands of mailers, catalogs and sales pitches go straight into the trash]$_{Arg2}$.

2 Previous Work

With full discolouse, I took full inspiration from a previous paper that I had read by the University of Brandeis by Rutherford and Xue, further readings of their paper can be found on the citation page. Their project was far more complex than this project. Their paper discussed how implicit relations are more difficult to detect than explicit relations. And understandably so given the nature of implicit relations. Rutherford and Xue wanted to increase the accuracy of implicit relations by supplementing their training data with more implicit relations. The bottleneck is the annotation. It's not as difficult to detect and thus annotate explicit relations, because they're explicit! Their solution was to extract explicit relations in another corpus and convert these to implicit relations by dropping the connective and then supplementing their training data with more implicit relations. This was complex, because the connectives must meet a certain set of criteria to be dropped and converted into an implicit relation. They hoped, by adding additional discourse annotations, that this would increase the overall performance of their classifier. While their paper focused on supplementing datasets, this project will focus on how exactly one builds a discourse relation classifier.

3 Approach

This project required a lot of self learning of external tools, libraries, and corpora. I first had to import and parse the penn discourse tree. This proved to be quite challenging. The corpus had a ton of useful metadata, but was in a very hectic and complex csv file. Luckily, a good samaritan on github had already developed a python library to easily extract the text, relation, and any other metadata needed from the corpus. Once this was completed, I cleaned the data. Specifically, I converted the text into lowercase, removed punctuations, stopwords, and numbers. Then I needed to convert the data into a format that the sklearn python library would accept. The format was text in an array, where the text was the first phrase, the connective, and the second phrase concatenated. The relations also had to be converted into an array. The processing and formatting of the corpus was complete on my end. I then split the text corpus and relation corpus into a training and testing set, 80/20 respectively. And like any learning model, the text needed to be converted into numbers. This was a fairly simple process as sklearn has a very simple function to convert text into a tf-dif representation. Once this step was complete, I fed the tf-dif matrix into the sklearn naive bayes learning model. The model was trained and then ran on the test dataset. The correctness of the model was then placed into a confusion matrix. From there, evaluation metrics (precision, accuracy, recall) were computed manually.

4 Results

Once the test results were put into a confusion matrix, the evaluation metric were simple to compute. I will not explain the calculations of the metrics, since it would be a very long and tedious explanation. But, the calculation work is shown in the tables below. Surprisingly, the explicit and implicit classifiers performed similarly for precision. Explicit relations scores a 58% while implicit scored just below this with a 57%. The entRel relation scored the highest precision with 62%. Explicit scored well above the rest of the classes with a 83% and makes sense since the learning model can pick up on explicit words. The other classes scored lower than 52% with the noRel relation scoring the lowest at 38%. The accuracy of the entire system was a 57%. This is surprising since the previous paper of Rutherford and Xue performed at an accuracy rate of 57%. And this is aftering adding their supplemental dataset. Discussion on why this maybe be is explored in the next section. I did however tweak some attributes to find the best performing system. Lemmatizing each word does not impact the system, but filtering out words with a frequency less than 4 significantly increased the performance by 7%, the previous accuracy being only 50% and 47% without pre processing. I had planned to run the system on sklearn's SVM learning model, but this was much more difficult anticipated and consequently discarded.

Confusion Matrix

	Explicit	**Implicit**	**AltLex**	**EntRel**	**NoRel**
Explicit	3038	616	0	25	0
Implicit	1631	1514	0	40	0
AltLex	72	66	0	0	0
EntRel	465	483	0	104	0
NoRel	25	35	0	6	0

Precision

Explicit	3038/(3038+1631+72+465+25) = **0.58**
Implicit	1541/(616+1514+66+483+35) = **0.57**

AltLex	0/0 = **0.00**
EntRel	104/(25+40+104) = **0.62**
NoRel	0/0 = **0.00**

Recall

Explicit	3038/(3038+616+25) = **0.83**
Implicit	1631/(1631+1514+40) = **0.51**
AltLex	72/(72+66) = **0.52**
EntRel	465/(465+483+104) = **0.44**
NoRel	25/(25+35+6) = **0.38**

Total Accuracy

Total Correct/ Total = **0.57**

Lincoln Samelson
12/19/2018
Ling 506
Final Project

5 Discussion

The most intriguing part of this project was how the system performed and exceeded my expectations. Without pre processing or tweaking any attributes, the system performed at an accuracy rate of 47%. With pre processing the system had an accuracy of 50%, which was exactly what Rutherford and Xue had prior to the supplemental dataset. However, the next accuracy of 57% could be suspect to many readers. But when I dropped all words below a frequency of 4, the system jumped to 57%. This is the same accuracy rate of Rutherford and Xue's system along with their supplemental data. So can we now conclude that adding supplemental data is just as useful as dropping small frequency words? Not entirely. For one, their system was built using a distant supervision model. My system used a naive bayes learning approach. Second, we have no way of knowing what exactly their pre processing stage was or their approach to filtering out low frequency words. Couple this with potential ametuer errors and the conclusion is unclear. If another experiment were conducted on this topic, I would urge the researchers to try and have a baseline model that does exactly what Rutherford and Xue did. Then can go from there and see how tweaking attributes or even running different models can affect the accuracy of a discourse relation classifying system.

Lincoln Samelson
12/19/2018
Ling 506
Final Project

Work Cited

Rutherford, Attapol, and Nianwen Xue. “Improving the Inference of Implicit Discourse Relations via Classifying Explicit Discourse Connectives.” *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, doi:10.3115/v1/n15-1081.