Shallow Discourse Parsing with Syntactic and (a Few) Semantic Features

Shubham Mukherjee, Abhishek Tiwari, Mohit Gupta and Anil Kumar Singh
Department of Computer Science and Engineering
Indian Institute of Technology (BHU), Varanasi, India
{shubham.mukherjee.cse11,abhishek.ktiwari.cse11,
mohit.gupta.cse11,aksingh.cse}@iitbhu.ac.in

Abstract

Discourse parsing is a challenging task and is crucial for discourse analysis. In this paper, we focus on labelling argument spans of discourse connectives and sense identification in the CoNLL-2015 shared task setting. We have used syntactic features and have also tried a few semantic features. We employ a pipeline of classifiers, where the best features and parameters were selected for each individual classifier, based on experimental evaluation. We could only get results somewhat better than of the baseline on the overall task, but the results over some of the sub-tasks are encouraging. Our initial efforts at using semantic features do not seem to help.

1 Introduction

Different natural language constructs are dependent on each other to form a coherent discourse. Extraction of discourse relations is a challenging task. Interest in discourse parsing has increased after the release of the Penn Discourse TreeBank (PDTB) (Miltsakaki et al., 2004). Shallow discourse parsing involves classifying connectives, relation classification and labelling argument spans, the last of which is considerably harder.

Previously, an end-to-end model by Lin et al. (2014) was developed which used only syntactic features from parse trees and improved the discourse parser performance. In our paper, we have constructed an analogous pipeline of classifiers which extracts the shallow discourse information based on the PDTB based annotation scheme. However, since discourse relations directly affect the semantic understanding of the text, the use of semantic features can prove useful if explored. We tried to use a few such features, though without much success. Implicit relations were handled using a heuristic-based baseline parser.

2 Resources and Corpus

For our purposes, we needed syntactic parse trees for the extraction of syntactic features, for which we used the PDTB corpus. These features were used for training each classifier stage of the pipeline.

The PDTB is the first large-scale corpus including a million words taken from the Wall Street Journal (Miltsakaki et al., 2004) and is based on the observation that no discourse relations in any language have been identified with more than two arguments. It uses the connective as the predicate, and the two text spans as the predicate's argument. Specifically, the span syntactically attached to the connective is Arg1 and the second span is Arg2.

The relative position of the Arg1 and Arg2 can appear in any order, at any distance to each other, although the position of Arg2 is fixed once the connective is identified in case of explicit relations. There are distribution statistics from (Miltsakaki et al., 2004) which will prove beneficial in our algorithm. For example, in explicit relations, Arg1 precedes Arg2 39.51% of times and lies in the same sentence 60.38% of the times. Even when Arg1 precedes Arg2, 79.9% of cases are with adjacent sentences. Also, almost all (96.8%) of the implicit cases are where Arg1 precedes Arg2.

For our experiments, we required syntactic features derived from parse trees, along with semantic features. Tokenisation was based on the gold standard PTB tree structure. The parsed trees, which were provided by CoNLL-2015 shared task organisers, were created by the Berkeley parser and were provided in the *json* format.

We did not use any other resources.

3 Related Work

Argument labelling can be done by locating parts within an argument, or by labelling the entire span, the latter being the preferred method. Explicit connective classification is usually done beforehand. (Pitler and Nenkova, 2009) achieved an F-Score of 94.19% which was extended by (Lin et al., 2014) to get an F-score of 95.36%.

Various approaches have been used for argument span labelling. (Ghosh et al., 2014) used a linear tagging approach based on Conditional Random Fields. (Lin et al., 2014), however, used a completely different approach using argument node classification within the syntax tree. Our approach resembles the architecture used by (Lin et al., 2014), with the addition of a few semantic features. Surprisingly, semantic features have not been tried for this task.

A hybrid approach was explored by (Kong et al., 2014) taking advantages of both the linear tagging and sub-tree extraction by using a constituent based approach. In contrast, we employ the idea of integrating additional heuristic and semantic features at different points in the pipeline.

For non-explicit relations, (Lin et al., 2009) have used word-pair features, which was the Cartesian product of all words from Arg1 and Arg2. Simple heuristic-based approaches have also shown reasonably high accuracy (Xue et al., 2015).

4 An Overview of Our Approach

Similar to the Lin et. al. (2014) model, we employed use a pipeline of classifiers, namely Explicit Connective Classifier, Argument Position Classifier, Argument Identifier and Sense Classifier. We used a seperate but linked parser for non-explicit cases. Given only the raw text of the sentence(s) and their parse trees, we attempt to determine:

1. Whether the sentence(s) have discourse relation present. And if so, the location of the connective in case of explicit relations.

2. The Argument span of the two arguments in terms of token numbers.

3. The sense of the discourse relation.

For this purpose, we employed a modular approach, building classifiers for each stage of the process. Each module effectively performs a classification task.

Each classifier was trained individually with the inclusion of heuristics and a variety of features. Evaluation after each modification enabled selection of better parameters for that particular module. In particular, the methodology used for connective matching and usage of the uncovered sets is explained in sections 5.1 and 8, respectively. We also explored the use of semantic features at each stage.

5 Explicit Connective Classification

Explicit connective classification involves two steps: (a) matching all the occurrences of the connective and (b) disambiguating them as discourse vs. non-discourse. We have used the set of 99 connective heads from the PDTB Annotation Manual[1] (2007) to match the connectives and then classified them based on the features extracted from the connective.

5.1 Connective Matching

The PDTB Annotation Manual gives an exhaustive list of 99 connective heads, based on which we generated rules for extracting occurrences of each of the connectives (such as *and*, *or*, *therefore* etc.). As a prepossessing step in the training dataset, the entire connective span was first mapped to the connective heads using a mapper script for cases like "either... or" and "if... then", which had to be treated separately by considering the entire segment as a whole. This method ensures that all connectives are identified exhaustively and the training process improves.

5.2 Features Used

(Lin et al., 2014) used an extension of the syntactic features used by (Pitler and Nenkova, 2009), which resulted in a higher F-score of 95.36%. They were extracted after generating appropriate graphical representation of the parse tree[2]. We employed the same features, along with a few semantic features (see section 7).

[1]`http://www.seas.upenn.edu/~pdtb/PDTBAPI/pdtb-annotation-manual.pdf`

[2]Python NLTK library was used for all syntax features in every module.

6 Argument Identification with Additional Heuristics

Argument identification is directly dependent on the relative position of the arguments as shown by (Lin et al., 2014). For Arg1 occurring before Arg2 (the PS class), a baseline parser with the assumption of adjacent sentences was used. This is motivated by the fact that 79.9% of cases within the PS class lie in this category. Extension of the same logic was used for non-explicit discourse relations as described in section 8. Where both arguments occur in the same sentence (the SS class) (Lin et al., 2014) used node classification with syntactic parse tree. In our system, we used additional heuristics observed from manual observations of incorrect cases.

After applying the implementation, we manually observed each case in the training set where there was a mismatch between the expected and predicted results. Observations showed that the Arg1 node tends to appear towards the root of the sentence encompassing the entire sentence. And the Arg2 node often tends to appear towards the leaf nodes.

Although in some cases the Arg1 node may indeed be the root of the syntax tree, but those cases were infrequent. Hence, after altering the algorithm to specifically avoid any of the above mentioned scenarios, the results showed marginal but noticeable improvement in both arg1 and arg2 performance. F-Score of the Arg1 node detection improved by 2.14% and the corresponding score for the Arg2 node improved by 0.1%.

7 Use of Semantic Features

All the related work referred to in this paper only used syntactic features for every classifier. Intuitively speaking, there seems to be a good case for using semantic features, in addition to syntactic features. However, the extraction of semantic features from raw text is comparatively hard.

Many semantic features also tend to be inconsistent or sparse. Features detected in one sentence may be completely absent in a large majority of sentences, leading to ineffective features.

7.1 The Boxer Tool

The C&C tool Boxer (Bos, 2014), developed by Johan Bos, is a toolkit for creating the semantic representation of sentences, developed by Johan Bos. Boxer is capable of extracting features from the majority of sentences, although it has some limitations. Boxer works by chunking the sentence into blocks or 'boxes' and then subsequently identifying semantic relations between them. The Boxer tool also marks the tokens of specific semantic roles such as agent, patient etc. We only used the features which were more commonly occurring. These were: the POS tag of the agent's token, the theme's token and the patient's token. We used the POS tags instead of the tokens themselves because tags are more general whereas tokens become too specific (with lower frequencies). Based on results and on more reflection we realize that this choice was not well motivated.

7.2 Application of Features

As mentioned in section 4, we tried the integration of semantic features at each feasible point in the pipeline and tested the results. Since labelling of Arg1 and Arg2 nodes is done through node-wise feature extraction, semantic features, which are extracted from a sentence as a whole, could not be easily integrated. Semantic features were included in two classifier stages: (a) the argument position classifier and (b) the sense classifier. This was the best combination we could get for semantic features. However, the integration of these additional features did not improve the overall performance. Instead, there was a 1.1% decrease in the overall parser F1 score. The inability of the semantic features to improve the classifier performance can be attributed to the fact that the particular features used had high correlation with syntactic features.

8 Non-explicit Identification

We have used a simple heuristic-based baseline parser as done by Lin et al. (2014) for implicit connectives. The parser is based on the adjacent sentence argument assumption mentioned in section 6. This is motivated by the fact that non-explicit relation also have a majority of cases in this category, akin to the PS case for explicit relations.

We used sets to mark the sentences in the same sentence category for explicit relations as covered. The rest of the sentences were marked as uncovered. Distinction between explicit and non-explicit cases were made while marking the sentences. The argument spans were then marked taking a pair of sentences as arguments, the sentence occurring earlier being Arg1. The explicit relation

Classifier	Precision	Recall	F1 Score	Type of Classifier
Connective Classifier	91.76%	91.70%	91.73%	Maximum-Entropy
Argument Position	98.79%	97.11%	97.94%	Maximum-Entropy
Argument Position + Semantic Features	97.61%	93.18%	95.34%	Naive-Bayesian
Argument Extraction (Arg1 + Arg2)	21.89%	34.47%	26.78%	Maximum-Entropy
Sense	29.95%	6.33%	5.95%	Maximum-Entropy
Sense + Semantic Features	27.81%	5.39%	4.68%	Naive-Bayesian

Table 1: Individual Classifier Analysis

	Dev Set			
Parameter	**Without Semantic Features**	**With Semantic Features**	**Blind Set**	**Test Set**
Arg 1 Arg2 extraction f1	26.78%	26.78%	21.71%	22.52%
Arg 1 Arg2 extraction precision	21.89%	21.89%	18.14%	18.19%
Arg 1 Arg2 extraction recall	34.46%	34.37%	27.05%	29.55%
Arg1 extraction f1	36.25%	36.25%	32.2%	32.7%
Arg1 extraction precision	29.63%	29.63%	26.9%	26.41%
Arg1 extraction recall	46.66%	46.66%	40.12%	42.91%
Arg2 extraction f1	49.82%	49.81%	48.87%	44.68%
Arg2 extraction precision	40.73%	40.73%	40.82%	36.1%
Arg2 extraction recall	64.14%	64.14%	60.88%	58.64%
Explicit connective f1	93.55%	93.55%	89.3%	93.06%
Explicit connective precision	95.41%	95.41%	91.67%	93.93%
Explicit connective recall	91.76%	91.76%	87.05%	92.2%
Sense f1	5.95%	4.68%	6.44%	7.17%
Sense precision	29.95%	27.81%	14.87%	25.67%
Sense recall	6.33%	5.39%	7.16%	8.05%
Overall Precision	7.21%	6.32%	6.38%	5.78%
Overall Recall	11.35%	9.96%	9.51%	9.39%
Overall F1	8.82%	7.74%	7.64%	7.15%

Table 2: Overall Parser Performance for Explicit Connectives

sentences among these were then sense classified, along with the PS category explicit sentences.

9 Explicit Sense Classification

Sense classification is the final step in our model. (Lin et al., 2014) reported an F-Score of 86.77% using connective-based features over the PDTB corpus. The integration of semantic features was done as described in section 7. This degraded the F-Score by 1.3%. Thus, the use of (the few) semantic features with high correlation to syntactic features decreases the performance.

10 Evaluation

The pipeline architecture which was used had several classifiers, each of which was evaluated individually on two kinds of training models: (a) the Naive-Bayesian classifier and (b) the Maximum Entropy classifier. For each of the individual classifiers, training and test sets were divided in a 4:1 ratio with a 5-fold cross-validation.

Table 1 presents the individual classifier results. Each phase was tested on the Naive-Bayesian and the Maximum Entropy classifiers. The better one for each sub-task is displayed. It should be noted that sense classification was computed only when both Arg1 and Arg2 spans exactly matched. Sense classification with incorrect argument spans would not be a useful statistical measure.

The overall parser performance was also measured when the complete end-to-end pipeline was implemented for sentences, accompanied by their corresponding syntactic parse trees and feature representations. Table 2 presents the overall best performance on the blind set after multiple trials on a sufficiently large subset of the PDTB corpus. It further compares performance of the overall system with semantic features included vis-a-vis without them. The overall parser performance is only somewhat better (nearly double) than those of the baseline, but the sense classification recall, sense precision and Arg1 extraction precision are bringing down the overall F1 score, as the performance on other sub-tasks is relatively much better. We are investigating the cause for this.

11 Conclusion

We used an end-to-end shallow discourse parser, which is an extension of the work described in (Lin et al., 2014), with the addition of some heuristics and a few semantic features obtained from the Boxer tool. The core idea is using syntactic and semantic features for classification and labelling. However, we were not able to get better results with the semantic features that we tried. We plan to explore more sophisticated semantic features. While our overall performance was relatively low, we did get good results for some of the sub-tasks. We will try to include more results in the final version of the paper.

References

Johan Bos. 2014. Open-domain semantic parsing with boxer. In *Proceedings of the 20th Nordic Conference of Computational Linguistics*, pages 301–304.

Sucheta Ghosh, Richard Johansson, Giuseppe Riccardi, and Sara Tonelli. 2014. Shallow discourse parsing with conditional random fields. In *Proceedings of the 5th International Joint Conference on Natural Language Processing.*

Fang Kong, Hwee Tou Ng, and Guodong Zhou. 2014. A constituent-based approach to argument labeling with joint inference in discourse parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing implicit discourse relations in the penn discourse treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing.*

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A pdtb-styled end-to-end discourse parser. *Natural Language Engineering, Cambridge Univ Press.*

Leni Miltsakaki, Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2004. The penn discourse treebank. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation.*

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers.*

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Rashmi Prasad, Christopher Bryant, and Attapol Rutherford. 2015. The conll-2015 shared task on shallow discourse parsing. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning: Shared Task.*