The Integration of Syntactic Parsing and Semantic Role Labeling

Szu-ting Yi
University of Pennsylvania
3330 Walnut Street
Philadelphia, PA 19104 USA
szuting@linc.cis.upenn.edu

Martha Palmer
University of Pennsylvania
3330 Walnut Street
Philadelphia, PA 19104 USA
mpalmer@linc.cis.upenn.edu

Abstract

This paper describes a system for the CoNLL-2005 Shared Task on Semantic Role Labeling. We trained two parsers with the training corpus in which the semantic argument information is attached to the constituent labels, we then used the resulting parse trees as the input of the pipelined SRL system. We present our results of combining the output of various SRL systems using different parsers.

1 Introduction

Semantic parsing, identifying and classifying the semantic entities in context and the relations between them, potentially has great impact on its downstream applications, such as text summarization, question answering, and machine translation. As a result, semantic parsing could be an important intermediate step for natural language comprehension. In this paper, we investigate the task of Semantic Role Labeling (SRL): Given a verb in a sentence, the goal is to locate the constituents which are arguments of the verb, and assign them appropriate semantic roles, such as, Agent, Patient, and Theme.

Previous SRL systems have explored the effects of using different lexical features, and experimented on different machine learning algorithms. (Gildea and Palmer, 2002; Pradhan et al., 2005; Punyakanok et al., 2004) However, these SRL systems generally extract features from sentences processed by a syntactic parser or other shallow parsing components, such as a chunker and a clause identifier. As a result, the performance of the SRL systems relies heavily on those syntax-analysis tools.

In order to improve the fundamental performance of an SRL system, we trained parsers with training data containing not only syntactic constituent information but also semantic argument information. The new parsers generate more correct constituents than that trained on pure syntactic information. Because the new parser generate different constituents than a pure syntactic parser, we also explore the possibility of combining the output of several parsers with the help of a voting post-processing component.

This paper is organized as follows: Section 2 demonstrates the components of our SRL system. We elaborate the importance of training a new parser and outline our approach in Section 3 and Section 4. Finally, Section 5 reports and discusses the results.

2 Semantic Role Labeling: the Architecture

Our SRL system has 5 phases: Parsing, Pruning, Argument Identification, Argument Classification, and Post Processing. The Argument Identification and Classification components are trained with Sec 02-21 of the Penn Treebank corpus.

2.1 Parsing

Previous SRL systems usually use a pure syntactic parser, such as (Charniak, 2000; Collins, 1999), to retrieve possible constituents. Once the boundary of a constituent is defined, there is no way to change it in later phases. Therefore the quality of the syntactic parser has a major impact on the final per-

Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL),
pages 237–240, Ann Arbor, June 2005.

formance of an SRL system, and the percentage of correct constituents that is generated by the syntactic parser also defines the recall upper bound of an SRL system. In order to attack this problem, in addition to Charniak's parser (Charniak, 2000), our system combine two parser which are trained on both syntactic constituent information and semantic argument information. (See Section 3)

2.2 Pruning

Given a parse tree, a pruning component filters out the constituents which are unlikely to be semantic arguments in order to facilitate the training of the Argument Identification component. Our system uses the heuristic rules introduced by (Xue and Palmer, 2004). The heuristics first spot the verb and then extract all the sister nodes along the verb spine of the parse tree. We expand the coverage by also extracting all the immediate children of an S, ADVP, PP and NP node. This stage generally prunes off about 80% of the constituents given by a parser. For our newly trained parsers, we also extract constituents which have a secondary constituent label indicating the constituent in question is an argument.

2.3 Argument Identification and Classification

We have as our Argument Identification component a binary maximum-entropy classifier to determine whether a constituent is an argument or not. If a constituent is tagged as an argument, the Argument Classification component, which is a multi-class maximum-entropy classifier, would assign it a semantic role. The implementation of both the Argument Identification and Classification components makes use of the Mallet package[1].

The lexical features we use to train these two components are taken from (Xue and Palmer, 2004).

We trained the Argument Identification component with the following single features: the **path** from the constituent to the verb, the **head word** of the constituent and its **POS tag**, and the **distance** between the verb and the constituent, and **feature combinations**: the verb and the phrasal type of the constituent, the verb and the head word of the constituent. If the parent node of the constituent is a PP node, then we also include the head word of the PP node and the feature combination of the verb and the head word of the PP node.

In addition to the features listed above, the Argument Classification component also contains the following features: the **verb**, the first and the last **content word** of the constituent, the **phrasal type** of the left sibling and the parent node, **voice** (passive or active), **position** of the constituent relative to the verb, the **subcategorization frame**, and the **syntactic frame** which describes the sequential pattern of the noun phrases and the verb in the sentence.

2.4 Post Processing

The post processing component merges adjacent discontinuous arguments and marks the R-arguments based on the content word and phrase type of the argument. Also it filters out arguments according to the following constraints:

1. There are no overlapping arguments.
2. There are no repeating core arguments.

In order to combine the different systems, we also include a voting scheme. The algorithm is straightforward: Suppose there are N participating systems, we pick arguments with N votes, N-1 votes ..., and finally 1 vote. The way to break a tie is based on the confidence level of the argument given by the system. Whenever we pick an argument, we need to check whether this argument conflicts with previously selected arguments based on the constraints described above.

3 Training a Parser with Semantic Argument Information

A good start is always important, especially for a successful SRL system. Instead of passively accepting candidate constituents from the upstream syntactic parser, an SRL system needs to interact with the parser in order to obtain improved performance. This motivated our first attempt which is to integrate syntactic parsing and semantic parsing as a single step, and hopefully as a result we would be able to discard the SRL pipeline. The idea is to augment the Penn Treebank (Marcus et al., 1994) constituent labels with the semantic role labels from the PropBank (Palmer et al., 2005), and generate a rich training corpus. For example, if an *NP* is also an ar-

[1] http://mallet.cs.umass.edu

gument *ARG0* of a verb in the given sentence, we change the constituent label *NP* into *NP-ARG0*. A parser therefore is trained on this new corpus and should be able to serve as an SRL system at the same time as predicting a parse.

However, this ideal approach is not feasible. Given the fact that there are many different semantic role labels and the same constituent can be different arguments of different verbs in the same sentence, the number of constituent labels will soon grow out of control and make the parser training computationally infeasible. Not to mention that anchor verb information has not yet been added to the constituent label, and general data sparseness. As a compromise, we decided to integrate only Argument Identification with syntactic parsing. We generated the training corpus by simply marking the constituents which are also semantic arguments.

4 Parsing Experiments

We trained a maximum-entropy parser based on (Ratnaparkhi, 1999) using the OpenNLP package [2]. We started our experiments with this specific parsing implementation because of its excellent flexibility that allows us to test different features. Besides, this parser contains four clear parse tree building stages: TAG, CHUNK, BUILD, and CHECK. This parsing structure offers us an isolated working environment for each stage that helps us confine necessary implementation modifications and trace down implementation errors.

4.1 Data Preparation

Following standard practice, we use Sec 02-21 of the Penn Treebank and the PropBank as our training corpus. The constituent labels defined in the Penn Treebank consist of a primary label and several secondary labels. A primary label represents the major syntactic function carried by the constituent, for instance, NP indicates a noun phrase and PP indicates a prepositional phrase. A secondary label, starting with "-", represents either a grammatical function of a constituent or a semantic function of an adjunct. For example, NP-SBJ means the noun phrase is a surface subject of the sentence; PP-LOC means the prepositional phrase is a location. Although the secondary labels give us much to encourage information, because of data sparseness problem and training efficiency, we stripped off all the secondary labels from the Penn Treebank.

After stripping off the secondary labels from the Penn Treebank, we augment the constituent labels with the semantic argument information from the PropBank. We adopted four different labels, -AN, -ANC, -AM, and -AMC. If the constituent in the Penn Treebank is a core argument, which means the constituent has one of the labels of ARG0-5 and ARGA in the PropBank, we attach -AN to the constituent label. The label -ANC means the constituent is a discontinuous core argument. Similarly, -AM indicates an adjunct-like argument, ARGM, and -AMC indicates a discontinuous ARM.

For example, the sentence from Sec 02, *[ARG0 The luxury auto maker] [ARGM-TMP last year] sold [ARG1 1,214 cars] [ARGM-LOC in the U.S.]*, would appear in the following format in our training corpus: *(S (NP-AN (DT The) (NN luxury) (NN auto) (NN maker)) (NP-AM (JJ last) (NN year)) (VP (VBD sold) (NP-AN (CD 1,214) (NNS cars)) (PP -AM (IN in) (NP (DT the) (NNP U.S.)))))*

4.2 The 2 Different Parsers

Since the core arguments and the ARGMs in the PropBank loosely correspond to the complements and adjuncts in the linguistics literature, we are interested in investigating their individual effect on parsing performance. We trained two parsers. An **AN**-parser was trained on the Penn Treebank corpus augmented with two semantic argument labels: -AN, and -ANC. Another **AM**-parser was trained on labels -AM, and -AMC.

5 Results and Discussion

Table 1 shows the results after combining various SRL systems using different parsers. In order to explore the effects of combining, we include the overall performance on the development dataset of individual SRL systems in Table 2.

The performance of Semantic Role Labeling (SRL) is determined by the quality of the syntactic information provided to the system. In this paper, we investigate that for the SRL task whether it is more suitable to use a parser trained with data con-

[2] http://sourceforge.net/projects/opennlp/

	Precision	Recall	$F_{\beta=1}$
Development	75.70%	69.99%	72.73
Test WSJ	77.51%	72.97%	75.17
Test Brown	67.88%	59.03%	63.14
Test WSJ+Brown	76.31%	71.10%	73.61

Test WSJ	Precision	Recall	$F_{\beta=1}$
Overall	77.51%	72.97%	75.17
A0	85.14%	77.32%	81.04
A1	77.61%	75.16%	76.37
A2	68.18%	62.16%	65.03
A3	66.91%	52.60%	58.90
A4	77.08%	72.55%	74.75
A5	100.00%	40.00%	57.14
AM-ADV	59.73%	51.58%	55.36
AM-CAU	67.86%	52.05%	58.91
AM-DIR	65.67%	51.76%	57.89
AM-DIS	80.39%	76.88%	78.59
AM-EXT	78.95%	46.88%	58.82
AM-LOC	57.43%	55.37%	56.38
AM-MNR	54.37%	56.10%	55.22
AM-MOD	96.64%	94.01%	95.31
AM-NEG	96.88%	94.35%	95.59
AM-PNC	41.38%	41.74%	41.56
AM-PRD	50.00%	20.00%	28.57
AM-REC	0.00%	0.00%	0.00
AM-TMP	77.13%	74.15%	75.61
R-A0	86.82%	85.27%	86.04
R-A1	67.72%	82.05%	74.20
R-A2	46.15%	37.50%	41.38
R-A3	0.00%	0.00%	0.00
R-A4	0.00%	0.00%	0.00
R-AM-ADV	0.00%	0.00%	0.00
R-AM-CAU	0.00%	0.00%	0.00
R-AM-EXT	0.00%	0.00%	0.00
R-AM-LOC	100.00%	42.86%	60.00
R-AM-MNR	33.33%	33.33%	33.33
R-AM-TMP	78.57%	63.46%	70.21
R-C-A1	0.00%	0.00%	0.00
V	97.35%	95.54%	96.44

Table 1: Overall results (top) and detailed results on the WSJ test (bottom).

taining both syntactic bracketing and semantic argument boundary information than a pure syntactic one.

The results of the SRL systems using the AM- or AN- parsers are not significantly better than that using the Charniak's parser. This might due to the simple training mechanism of the base parsing algorithm which the AM- and AN- parsers exploit. It also suggests our future work to apply the approach to more sophisticated parsing frameworks. By then, We show that we can boost the final performance by combining different SRL systems using different parsers, given that the combination algorithm is capable of maintaining the quality of the final arguments.

	Precision	Recall	$F_{\beta=1}$
AN-parser	71.31%	63.68%	67.28
AM-parser	74.09%	65.11%	69.31
Charniak	76.31%	64.62%	69.98
All 3 combined	75.70%	69.99%	72.73

Table 2: Overall results on the development set of individual SRL systems.

6 Acknowledgments

We thank Tom Morton for providing detailed explanation for any of our parsing related inquiries.

References

Eugene Charniak. 2000. A Maximum-Entropy-Inspired Parser. In *Proceedings of NAACL-2000*.

Michael Collins. 1999. Head-Driven Statistical Models for Natural Language Parsing. *PhD Dissertation*, University of Pennsylvania.

Daniel Gildea and Martha Palmer. 2002. The Necessity of Parsing for Predicate Argument Recognition. In *Proceedings of ACL* 2002, Philadelphia, USA.

Mitchell Marcus, Grace Kim, Mary AnnMarcinkiewicz, et al. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In *Proceedings of ARPA Speech and Natural Language Workshop*.

Martha Palmer, Dan Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1).

Pradhan, S., Hacioglu, K., Krugler, V., Ward, W., Martin, J., and Jurafsky, D. 2005. Support Vector Learning for Semantic Argument Classification. To appear in *Machine Learning* journal, Special issue on Speech and Natural Language Processing.

V. Punyakanok, D. Roth, W. Yih, and D. Zimak. 2004. Semantic Role Labeling via Integer Linear Programming Inference. In *Proceedings of COLING*.

Adwait Ratnaparkhi. 1999. Learning to Parse Natural Language with Maximum Entropy Models. *Machine Learning*, 34, 151–175.

Nianwen Xue and Martha Palmer. 2004. Calibrating Features for Semantic Role Labeling. In *Proceedings of EMNLP*.