Semantic Role Labeling by Tagging Syntactic Chunks*

Kadri Hacioglu[1], **Sameer Pradhan**[1], **Wayne Ward**[1], **James H. Martin**[1], **Daniel Jurafsky**[2]
[1]University of Colorado at Boulder, [2]Stanford University
{hacioglu,spradhan,whw}@cslr.colorado.edu, martin@cs.colorado.edu, jurafsky@stanford.edu

Abstract

In this paper, we present a semantic role labeler (or chunker) that groups syntactic chunks (i.e. base phrases) into the arguments of a predicate. This is accomplished by casting the semantic labeling as the classification of syntactic chunks (e.g. NP-chunk, PP-chunk) into one of several classes such as the beginning of an argument (B-ARG), inside an argument (I-ARG) and outside an argument (O). This amounts to tagging syntactic chunks with semantic labels using the IOB representation. The chunker is realized using support vector machines as one-versus-all classifiers. We describe the representation of data and information used to accomplish the task. We participate in the "closed challenge" of the CoNLL-2004 shared task and report results on both development and test sets.

1 Introduction

In semantic role labeling the goal is to group sequences of words together and classify them by using semantic labels. For meaning representation the predicate-argument structure that exists in most languages is used. In this structure a word (most frequently a verb) is specified as a predicate, and a number of word groups are considered as arguments accompanying the word (or predicate).

In this paper, we select support vector machines (SVMs) (Vapnik, 1995; Burges, 1998) to implement the semantic role classifiers, due to their ability to handle an extremely large number of (overlapping) features with quite strong generalization properties. Support vector machines for semantic role chunking were first used in (Hacioglu and Ward, 2003) as word-by-word (W-by-W) classifiers. The system was then applied to the constituent-by-constituent (C-by-C) classification in (Hacioglu et al., 2003). In (Pradhan et al., 2003; Pradhan et al., 2004), several extensions to the basic system have been proposed, extensively studied and systematically compared to other systems. In this paper, we implement a system that classifies syntactic chunks (i.e. base phrases) instead of words or the constituents derived from syntactic trees. This system is referred to as the phrase-by-phrase (P-by-P) semantic role classifier. We participate in the "closed challenge" of the CoNLL-2004 shared task and report results on both development and test sets. A detailed description of the task, data and related work can be found in (Carreras and Màrquez, 2004).

2 System Description

2.1 Data Representation

In this paper, we change the representation of the original data as follows:

- Bracketed representation of roles is converted into IOB2 representation (Ramhsaw and Marcus, 1995; Sang and Veenstra, 1995)
- Word tokens are collapsed into base phrase (BP) tokens.

Since the semantic annotation in the PropBank corpus does not have any embedded structure there is no loss of information in the first change. However, this results in a simpler representation with a reduced set of tagging labels. In the second change, it is possible to miss some information in cases where the semantic chunks do not align with the sequence of BPs. However, in Section 3.2 we show that the loss in performance due to the misalignment is much less than the gain in performance that can be achieved by the change in representation.

*This research was partially supported by the ARDA AQUAINT program via contract OCG4423B and by the NSF via grant IIS-9978025

Sales	NNS	B−NP	(S*	−	(A1*A1)
declined	VBD	B−VP	*	decline	(V*V)
10	CD	B−NP	*	−	(A2*
%	NN	I−NP	*	−	*A2)
to	TO	B−PP	*	−	*
$	$	B−NP	*	−	(A4*
251.2	CD	I−NP	*	−	*
million	CD	I−NP	*	−	*A4)
from	IN	B−PP	*	−	*
$	$	B−NP	*	−	(A3*
278.7	CD	I−NP	*	−	*
million	CD	I−NP	*	−	*A3)
.	.	O	*S)	−	*

(a)

NP	Sales	NNS	B−NP	(S*	−	B−A1
VP	declined	VBD	B−VP	*	decline	B−V
NP	%	NN	I−NP	*	−	B−A2
PP	to	TO	B−PP	*	−	O
NP	million	CD	I−NP	*	−	B−A4
PP	from	IN	B−PP	*	−	O
NP	million	CD	I−NP	*	−	B−A3
O	.	.	O	*S)	−	O

(b)

Figure 1: Illustration of change in data representation; (a) original word-by-word data representation (b) phrase-by-phrase data representation used in this paper. Words are collapsed into base phrase types retaining only headwords with their respective features. Bracketed representation of semantic role labels is converted into IOB2 representation. See text for details.

The new representation is illustrated in Figure 1 along with the original representation. Comparing both we note the following differences and advantages in the new representation:

- BPs are being classified instead of words.
- Only the BP headwords (rightmost words) are retained as word information.
- The number of tagging steps is smaller.
- A fixed context spans a larger segment of a sentence.

Therefore, the P-by-P semantic role chunker classifies larger units, ignores some of the words, uses a relatively larger context for a given window size and performs the labeling faster.

2.2 Features

The following features, which we refer to as the base features, are provided in the shared task data for each sentence;

- **Words**
- **Predicate lemmas**
- **Part of Speech tags**
- **BP Positions**: The position of a token in a BP using the IOB2 representation (e.g. B-NP, I-NP, O etc.)
- **Clause tags**: The tags that mark token positions in a sentence with respect to clauses. (e.g *S)*S) marks a position that two clauses end)
- **Named entities**: The IOB tags of named entities. There are four categories; LOC, ORG, PERSON and MISC.

Using available information we have created the following token level features:

- **Token Position**: The position of the phrase with respect to the predicate. It has three values as "before", "after" and "-" for the predicate.
- **Path**: It defines a flat path between the token and the predicate as a chain of base phrases. At both ends, the chain is terminated with the POS tags of the predicate and the headword of the token.
- **Clause bracket patterns**: We use two patterns of clauses for each token. One is the clause bracket chain between the token and the predicate, and the other is from the token to sentence begin or end depending on token's position with respect to the predicate.
- **Clause Position**: a binary feature that indicates the token is inside or outside of the clause which contains the predicate
- **Headword suffixes**: suffixes of headwords of length 2, 3 and 4.
- **Distance**: we have two notions of distance; the first is the distance of the token from the predicate as a number of base phrases, and the second is the same distance as the number of VP chunks.

- **Length**: the number of words in a token.

We also use some sentence level features:

- **Predicate POS tag**: the part of speech category of the predicate
- **Predicate Frequency**; this is a feature which indicates whether the predicate is frequent or rare with respect to the training set. The threshold on the counts is currently set to 3.
- **Predicate BP Context** : The chain of BPs centered at the predicate within a window of size -2/+2.
- **Predicate POS Context** : The POS tags of the words that immediately precede and follow the predicate. The POS tag of a preposition is replaced with the preposition itself.
- **Predicate Argument Frames**: We used the left and right patterns of the core arguments (A0 through A5) for each predicate . We used the three most frequent argument frames for both sides depending on the position of the token in focus with respect to the predicate. (e.g. *raise* has A0 and A1_AO (A0 being the most frequent) as its left argument frames, and A1, A1_A2 and A2 as the three most frequent right argument frames)
- **Number of predicates**: This is the number of predicates in the sentence.

For each token (base phrase) to be tagged, a set of ordered features is created from a fixed size context that surrounds each token. In addition to the above features, we also use previous semantic IOB tags that have already been assigned to the tokens contained in the context. A 5-token sliding window is used for the context. A greedy left-to-right tagging is performed.

All of the above features are designed to implicitly capture the patterns of sentence constructs with respect to different word/predicate usages and senses. We acknowledge that they significantly overlap and extensive experiments are required to determine the impact of each feature on the performance.

2.3 Classifier

All SVM classifiers were realized using TinySVM[1] with a polynomial kernel of degree 2 and the general purpose SVM based chunker YamCha [2]. SVMs were trained for begin (B) and inside (I) classes of all arguments and one outside (O) class for a total of 78 one-vs-all classifiers (some arguments do not have an I-tag).

[1]http://cl.aist-nara.ac.jp/taku-ku/software/TinySVM

[2]http://cl.aist-nara.ac.jp/taku-ku/software/yamcha

Table 1: Comparison of W-by-W and P-by-P methods. Both systems use the base features provided (i.e. no feature engineering is done). Results are on dev set.

Method	Precision	Recall	$F_{\beta=1}$
P-by-P	69.04%	54.68%	61.02
W-by-W	68.34%	45.16%	54.39

Table 2: Number of sentences and unique training examples in each method.

Method	Sentences	Training Examples
P-by-P	19K	347K
W-by-W	19K	534K

3 Experimental Results

3.1 Data and Evaluation Metrics

The data provided for the shared task is a part of the February 2004 release of the PropBank corpus. It consists of sections from the Wall Street Journal part of the Penn Treebank. All experiments were carried out using Sections 15-18 for training Section-20 for development and Section-21 for testing. The results were evaluated for precision, recall and $F_{\beta=1}$ numbers using the *srl-eval.pl* script provided by the shared task organizers.

3.2 W-by-W and P-by-P Experiments

In these experiments we used only the base features to compare the two approaches. Table 1 illustrates the overall performance on the dev set. Although both systems were trained using the same number of sentences, the actual number of training examples in each case were quite different. Those numbers are presented in Table 2. It is clear that P-by-P method uses much less data for the same number of sentences. Despite this we particularly note a considerable improvement in recall. Actually, the data reduction was not without a cost. Some arguments have been missed as they do not align with the base phrase chunks due to inconsistencies in semantic annotation and due to errors in automatic base phrase chunking. The percentage of this misalignment was around 2.5% (over the dev set). We observed that nearly 45% of the mismatches were for the "outside" chunks. Therefore, sequences of words with outside tags were not collapsed.

3.3 Best System Results

In these experiments all of the features described earlier were used with the P-by-P system. Table 3 presents our best system performance on the development set. Additional features have improved the performance from 61.02 to 71.72. The performance of the same system on the test set is similarly illustrated in Table 4.

Table 3: System results on development set.

	Precision	Recall	$F_{\beta=1}$
Overall	74.17%	69.42%	71.72
A0	82.86%	78.50%	80.62
A1	72.82%	73.97%	73.39
A2	60.16%	56.18%	58.10
A3	59.66%	47.65%	52.99
A4	83.21%	74.15%	78.42
A5	100.00%	75.00%	85.71
AM-ADV	52.52%	41.48%	46.35
AM-CAU	61.11%	41.51%	49.44
AM-DIR	47.37%	15.00%	22.78
AM-DIS	76.47%	76.47%	76.47
AM-EXT	74.07%	40.82%	52.63
AM-LOC	51.21%	46.09%	48.51
AM-MNR	51.04%	36.83%	42.78
AM-MOD	99.47%	95.63%	97.51
AM-NEG	99.20%	94.66%	96.88
AM-PNC	70.00%	28.00%	40.00
AM-PRD	0.00%	0.00%	0.00
AM-REC	0.00%	0.00%	0.00
AM-TMP	69.33%	58.37%	63.38
R-A0	91.55%	80.25%	85.53
R-A1	72.46%	67.57%	69.93
R-A2	100.00%	52.94%	69.23
R-AM-LOC	100.00%	25.00%	40.00
R-AM-TMP	0.00%	0.00%	0.00
V	99.05%	99.05%	99.05

Table 4: System results on test set.

	Precision	Recall	$F_{\beta=1}$
Overall	72.43%	66.77%	69.49
A0	82.93%	79.88%	81.37
A1	71.92%	71.33%	71.63
A2	49.37%	49.30%	49.33
A3	57.50%	46.00%	51.11
A4	87.10%	54.00%	66.67
A5	0.00%	0.00%	0.00
AM-ADV	53.36%	38.76%	44.91
AM-CAU	57.89%	22.45%	32.35
AM-DIR	37.84%	28.00%	32.18
AM-DIS	66.83%	62.44%	64.56
AM-EXT	70.00%	50.00%	58.33
AM-LOC	46.63%	36.40%	40.89
AM-MNR	50.31%	31.76%	38.94
AM-MOD	98.12%	92.88%	95.43
AM-NEG	91.11%	96.85%	93.89
AM-PNC	52.00%	15.29%	23.64
AM-PRD	0.00%	0.00%	0.00
AM-TMP	64.57%	50.74%	56.82
R-A0	90.21%	81.13%	85.43
R-A1	83.02%	62.86%	71.54
R-A2	100.00%	33.33%	50.00
R-A3	0.00%	0.00%	0.00
R-AM-LOC	0.00%	0.00%	0.00
R-AM-MNR	0.00%	0.00%	0.00
R-AM-PNC	0.00%	0.00%	0.00
R-AM-TMP	60.00%	21.43%	31.58
V	98.46%	98.46%	98.46

4 Conclusions

We have described a semantic role chunker using SVMs. The chunking method has been based on a chunked sentence structure at both syntactic and semantic levels. We have jointly performed semantic chunk segmentation and labeling using a set of one-vs-all SVM classifiers on a phrase-by-phrase basis. It has been argued that the new representation has several advantages as compared to the original representation. It yields a semantic role labeler that classifies larger units, exploits relatively larger context, uses less data (possibly, redundant and noisy data are filtered out), runs faster and performs better.

References

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 Shared Task: Semantic Role Labeling *in the same volume of Proc. of CoNLL'2004 Shared Task*.

Christopher J. C. Burges. 1997. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2), pages 1-47.

Kadri Hacioglu and Wayne Ward. 2003. Target word Detection and Semantic Role Chunking Using Support Vector Machines. *Proc. of the HLT-NAACL-03*.

Kadri Hacioglu, Sameer Pradhan, Wayne Ward, James Martin, and Dan Jurafsky. 2003. Shallow Semantic Parsing Using Support Vector Machines. CSLR Tech. Report, CSLR-TR-2003-1.

Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James Martin, and Dan Jurafsky. 2003. Semantic Role Parsing: Adding Semantic Structure to Unstructured Text. *Proc. of Int. Conf. on Data Mining (ICDM03)*.

Sameer Pradhan, Kadri Hacioglu, Wayne Ward, James Martin, and Dan Jurafsky. 2004. Support Vector Learning for Semantic Argument Classification. *to appear in Journal of Machine Learning*.

Lance E. Ramhsaw and Mitchell P. Marcus. 1995. Text Chunking Using Transformation Based Learning. *Proc. of the 3rd ACL Workshop on Very Large Corpora*, pages 82-94.

Erik F. T. J. Sang, John Veenstra. 1999. Representing Text Chunks. *Proc. of EACL'99*, pages 173-179.

Vladamir Vapnik 1995. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, USA.