Semantic Role Labeling Using Support Vector Machines

Tomohiro Mitsumori[1]**, Masaki Murata**[2]**, Yasushi Fukuda**[3]
Kouichi Doi[1]**, and Hirohumi Doi**[1]

[1] Graduate School of Information Science, Nara Institute of Science and Technology

8916-5, Takayama-cho, Ikoma-shi, Nara, 630-0101, Japan

{mitsumor,doy}@is.naist.jp, doi@cl-sciences.co.jp

[2] National Institute of Information and Communications Technology

3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0289, Japan

murata@nict.go.jp

[3] Sony-Kihara Research Center Inc.

1-14-10 Higashigotanda, Shinagawa-ku, Tokyo, 141-0022, Japan

yasu@krc.sony.co.jp

Abstract

In this paper, we describe our systems for the CoNLL-2005 shared task. The aim of the task is semantic role labeling using a machine-learning algorithm. We apply the Support Vector Machines to the task. We added new features based on full parses and manually categorized words. We also report on system performance and what effect the newly added features had.

1 Introduction

The CoNLL-2005 shared task (Carreras and Màrquez, 2005) concerns the recognition of automatic semantic roles for the English language. Given a sentence, the task consists of analyzing the propositions expressed by various target verbs of the sentence. The semantic roles of constituents of the sentence are extracted for each target verb. There are semantic arguments such as Agent, Patient, and Instrument and also adjuncts such as Locative and Temporal. We performed the semantic role labeling using Support Vector Machines (SVMs). Systems that used SVMs achieved good performance in the CoNLL-2004 shared task, and we added data on full parses to it. We prepared a feature used by the full parses, and we also categorized words that appeared in the training set and added them as features. Here, we report on systems for automatically labeling semantic roles in a closed challenge in the CoNLL-2005 shared task.

This paper is arranged as follows. Section 2 describes the SVMs. Our system is described Sec-

tion 3, where we also describe methods of data representation, feature coding, and the parameters of SVMs. The experimental results and conclusion are presented in Sections 4 and 5.

2 Support Vector Machines

SVMs are one of the binary classifiers based on the maximum margin strategy introduced by Vapnik (Vapnik, 1995). This algorithm has achieved good performance in many classification tasks, e.g. named entity recognition and document classification. There are some advantages to SVMs in that (i) they have high generalization performance independent of the dimensions of the feature vectors and (ii) learning with a combination of multiple features is possible by using the polynomial kernel function (Yamada and Matsumoto, 2003). SVMs were used in the CoNLL-2004 shred task and achieved good performance (Hacioglu et al., 2004) (Kyung-Mi Park and Rim, 2004). We used YamCha (Yet Another Multipurpose Chunk Annotator)[1] (Kudo and Matsumoto, 2001), which is a general purpose SVM-based chunker. We also used TinySVM[2] as a package for SVMs.

3 System Description

3.1 Data Representation

We changed the representation of original data according to Hacioglu et al. (Hacioglu et al., 2004) in our system.

---

[1] http://chasen.org/~ taku/software/yamcha/
[2] http://chasen.org/~ taku/software/TinySVM/

- Bracketed representation of roles was converted into IOB2 representation (Ramhsaw and Marcus, 1995) (Sang and Veenstra, 1999).

- Word-by-word was changed to the phrase-by-phrase method (Hacioglu et al., 2004).

Word tokens were collapsed into base phrase (BP) tokens. The BP headwords were rightmost words. Verb phrases were not collapsed because some included more the one predicate.

3.2 Feature Coding

We prepared the training and development set by using files corresponding to: words, predicated partial parsing (part-of-speech, base chunks), predicate full parsing trees (Charniak models), and named entities. We will describe feature extraction according to Fig. 1. Figure 1 shows an example of an annotated sentence.

1st Words (Bag of Words): All words appearing in the training data.

2nd Part of Speech (POS) Tags

3rd Base Phrase Tags: Partial parses (chunks + clauses) predicted with UPC processors.

4th Named Entities

5th Token Depth : This means the degree of depth from a predicate (see Fig. 2). We used full parses predicted by the Charniak parser. In this figure, the depth of paid , which is a predicate, is zero and the depth of April is -2.

6th Words of Predicate

7th Position of Tokens: The position of the current word from the predicate. This has three value of "before", "after", and "-" (for the predicate).

8th Phrase Distance on Flat Path: This means the distance from the current token to the predicate as a number of the phrase on flat path. For example, the phrase distance of "April" is 4, because two "NP" and one "PP" exist from "paid"(predicate) to "April" (see 3rd column in Fig.1).

Table 1: Five most frequently categorized BP headwords appearing in training set.

Class	Examples
Person	he, I, people, investors, we
Organization	company, Corp., Inc., companies, group
Time	year, years, time, yesterday, months
Location	Francisco, York, California, city, America
Number	%, million, billion, number, quarter
Money	price, prices, cents, money, dollars

9th Flat Path: This means the path from the current word to the predicate as a chain of the phrases. The chain begins from the BP of the current word to the BP of the predicate.

10th Semantic Class : We collected the most frequently occurring 1,000 BP headwords appearing in the training set and tried to manually classified. The five classes (person, organization, time, location, number and money) were relatively easy to classify. In the 1,000 words, the 343 words could be classified into the five classes. Remainder could not be classified. The details are listed in Table 1.

Preceding class: The class (e.g. B-A0 or I-A1) of the token(s) preceding the current token. The number of preceding tokens is dependent on the window size. In this paper, the left context considered is two.
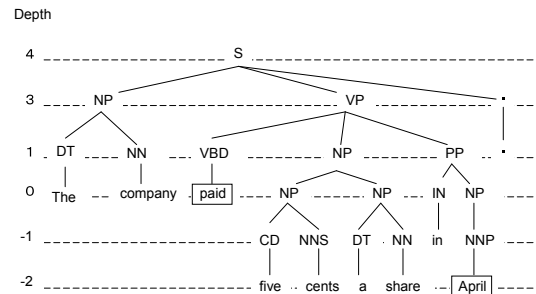

Figure 2: Parsing results obtained with Charniak parser and token depth.

3.3 Machine learning with YamCha

YamCha (Kudo and Matsumoto, 2001) is a general purpose SVM-based chunker. After inputting the training and test data, YamCha converts them for

198

(1st)	(2nd)	(3rd)	(4th)	(5th)	(6th)	(7th)	(8th)	(9th)	(10th)	(11th)
The	DT	B-NP	O	0	pay	before	-1	NP|VP	-	B-A0
company	NN	I-NP	O	0	pay	before	-1	NP|VP	organization	I-A0
paid	VBD	B-VP	O	0	pay	-	0	-	-	B-V
five	CD	B-NP	O	2	pay	after	1	VP|NP	-	B-A1
cents	NNS	I-NP	O	2	pay	after	1	VP|NP	money	I-A1
a	DT	B-NP	O	2	pay	after	2	VP|NP	-	I-A1
share	NN	I-NP	O	2	pay	after	2	VP|NP	-	I-A1
in	IN	B-PP	O	1	pay	after	3	VP|NP|PP	-	B-AM-TMP
April	NNP	B-NP	O	2	pay	after	4	VP|NP|PP|NP	time	I-AM-TMP
.	.	O	O	-1	pay	after	4	VP|NP|PP|NP|O	-	O

Data representation ⇩

company	NN	B-NP	O	0	pay	before	-1	NP|VP	organization	B-A0
paid	VBD	B-VP	O	0	pay	-	0	-	-	B-V
cents	NNS	B-NP	O	2	pay	after	1	VP|NP	money	B-A1
share	NN	B-NP	O	2	pay	after	2	VP|NP	-	I-A1
in	IN	B-PP	O	1	pay	after	3	VP|NP|PP	-	B-AM-TMP
April	NNP	B-NP	O	2	pay	after	4	VP|NP|PP|NP	time	I-AM-TMP
.	.	O	O	-1	pay	after	4	VP|NP|PP|NP|O	-	O

Figure 1: Example annotated sentence. Input features are words (1st), POS tags (2nd), base phrase chunks (3rd), named entities (4th), token depth (5th), predicate (6th), position of tokens (7th), phrase distance (8th), flat paths (9th), semantic classes (10th), argument classes (11th).

the SVM. The YamCha format for an example sentence is shown in Fig. 1. Input features are written in each column as words (1st), POS tags (2nd), base phrase chunks (3rd), named entities (4th), token depth (5th), predicate (6th), the position of tokens (7th), the phrase distance (8th), flat paths (9th), semantic classes (10th), and argument classes (11th). The boxed area in Fig. 1 shows the elements of feature vectors for the current word, in this case "share". The information from the two preceding and two following tokens is used for each vector.

3.4 Parameters of SVM

- Degree of polynomial kernel (natural number): We can only use a polynomial kernel in YamCha. In this paper, we adopted the degree of two.

- Range of window (integer): The SVM can use the information on tokens surrounding the token of interest as illustrated in Fig. 1. In this paper, we adopted the range from the left two tokens to the right two tokens.

- Method of solving a multi-class problem: We adopted the one-vs.-rest method. The BIO class is learned as (B vs. other), (I vs. other), and (O vs. other).

- Cost of constraint violation (floating number): There is a trade-off between the training error and the soft margin for the hyper plane. We adopted a default value (1.0).

4 Results

4.1 Data

The data provided for the shared task consisted of sections from the Wall Street Journal (WSJ) part of Penn TreeBank II. The training set was WSJ Sections 02-21, the development set was Section 24, and the test set was Section 23 with the addition of fresh sentences. Our experiments were carried out using Sections 15-18 for the training set, because the entire file was too large to learn.

4.2 Experiments

Our final results for the CoNLL-2005 shared task are listed in Table 2. Our system achieved 74.15% precision, 68.25% recall and 71.08 $F_{\beta=1}$ on the overall results of Test WSJ. Table 3 lists the effects of the token-depth and semantic-class features. The token-depth feature had a larger effect than that for the semantic class.

	Precision	Recall	$F_{\beta=1}$
Development	71.68%	64.93%	68.14
Test WSJ	74.15%	68.25%	71.08
Test Brown	63.24%	54.20%	58.37
Test WSJ+Brown	72.77%	66.37%	69.43

Test WSJ	Precision	Recall	$F_{\beta=1}$
Overall	74.15%	68.25%	71.08
A0	81.38%	76.93%	79.09
A1	73.16%	70.87%	72.00
A2	64.53%	59.01%	61.65
A3	61.16%	42.77%	50.34
A4	68.18%	58.82%	63.16
A5	100.00%	80.00%	88.89
AM-ADV	55.09%	43.87%	48.84
AM-CAU	60.00%	28.77%	38.89
AM-DIR	45.10%	27.06%	33.82
AM-DIS	72.70%	69.06%	70.83
AM-EXT	70.59%	37.50%	48.98
AM-LOC	55.62%	50.41%	52.89
AM-MNR	51.40%	42.73%	46.67
AM-MOD	97.04%	95.28%	96.15
AM-NEG	96.92%	95.65%	96.28
AM-PNC	56.00%	36.52%	44.21
AM-PRD	0.00%	0.00%	0.00
AM-REC	0.00%	0.00%	0.00
AM-TMP	73.39%	62.93%	67.76
R-A0	81.31%	71.88%	76.30
R-A1	59.69%	49.36%	54.04
R-A2	60.00%	18.75%	28.57
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	85.71%	28.57%	42.86
R-AM-MNR	100.00%	16.67%	28.57
R-AM-TMP	72.34%	65.38%	68.69
V	97.55%	96.05%	96.80

Table 2: Overall results (top) and detailed results on the WSJ test (bottom).

5 Conclusion

This paper reported on semantic role labeling using SVMs. Systems that used SVMs achieved good performance in the CoNLL-2004 shared task, and we added data on full parses to it. We applied a token-depth feature to SVM learning and it had a large effect. We also added a semantic-class feature and it had a small effect. Some classes were similar to the named entities, e.g., the PERSON or LOCATION of the named entities. Our semantic class feature also included not only proper names but also common words. For example, "he" and "she" also included the PERSON semantic class. Furthermore, we added a time, number, and money class. The

Table 3: Effects Token Depth (TD) and Semantic Class (SC) had on feature development set.

	Precision	Recall	$F_{\beta=1}$
Without DF and SC	68.07%	59.71%	63.62
With DF	71.36%	64.13%	67.55
With DF and SC	71.68%	64.93%	68.14

semantic class feature was manually categorized on the most frequently occurring 1,000 words in the training set. More effort of the categorization may improve the performance of our system.

References

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling . In *Proceedings of CoNLL-2005*.

Kadri Hacioglu, Sameer Pradhan, Wayne Ward, James H. Martin, and Daniel Jurafskey. 2004. Semantic Role Labeling by Tagging Syntactic Chunks. In *Proceedings of Conference on Computational Natural Language Learning (CoNLL-2004) Shared Task on Semantic Role Labeling*.

Taku Kudo and Yuji Matsumoto. 2001. Chunking with Support Vector Machines. In *Proceedings of Second Meeting of North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 192–199.

Young-Sook Hwang Kyung-Mi Park and Hae-Chang Rim. 2004. Semantic Role Labeling by Tagging Syntactic Chunks. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL-2004) Shared Task on Semantic Role Labeling*.

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

Erik F. T. J. Sang and John Veenstra. 1999. Representing Text Chunks. In *Proceedings of EACL !99*, pages 173–179.

Vladimir N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with Support Vector Machines . In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 195–206.