Semantic Role Labeling via Consensus in Pattern-Matching

Chi-San (Althon) Lin
Department of Computer Science
Waikato University
Hamilton, New Zealand
cl123@cs.waikato.ac.nz

Tony C. Smith
Department of Computer Science
Waikato University
Hamilton, New Zealand
tcs@cs.waikato.ac.nz

Abstract

This paper describes a system for semantic role labeling for the CoNLL2005 Shared task. We divide the task into two sub-tasks: boundary recognition by a general tree-based predicate-argument recognition algorithm to convert a parse tree into a flat representation of all predicates and their related boundaries, and role labeling by a consensus model using a pattern-matching framework to find suitable roles for core constituents and adjuncts. We describe the system architecture and report results for the CoNLL2005 development dataset.

1 Introduction

Semantic role labeling is to find all arguments for all predicates in a sentence, and classify them by semantic roles such as A0, A1, AM-TMP and so on. The performance of semantic role labeling can play a key role in Natural Language Processing applications, such as Information Extraction, Question Answering, and Summarization (Pradhan et al., 2004).

Most existing systems separate semantic role labeling into two sub-problems, boundary recognition and role classification, and use feature-based models to address both (Carreras et al., 2004). Our strategy is to develop a boundary analyzer by a general tree-based predicate-argument recognition algorithm (GT-PARA) for boundary recognition, and a pattern-matching model for role classification. The only information used in our system is Charniak's annotation with words, which contains all useful syntactic annotations. Five features, which are Headword, Phrase type, Voice, Target verb, and Preposition (of the first word), and a Pattern set, which includes numbers and types of roles in a pattern, are used for the pattern-matching approach. We develop a Pattern Database, trained by Wall Street Journal section 02 to 21, as our knowledge/Data base. The system outline is described in the following section.

2 System Description

An overview of the system architecture is shown in Figure 1. The input is a full parse tree for each sentence. We convert a sentence with words, and Charniak's information into a parsed tree as the input of GT-PARA. GT-PARA then converts the parse tree into a flat representation with all predicates and arguments expressed in [**GPLVR**] format; where

G: Grammatical function – 5 denotes subject, 3 object, and 2 others;
P: Phrase type of this boundary – 00 denotes ADJP, 01 ADVP, 02 NP, 03 PP, 04 S, 05 SBAR, 06 SBARQ, 07 SINV, 08 SQ, 09 VP, 10 WHADVP, 11 WHNP, 12 WHPP, and 13 Others
L: Distance (and position) of the argument with respect to the predicate that follows
V: Voice of the predicate, 0: active 1: passive
R: Distance (and position) of the argument with respect to the preceding predicate (n.b. **L** and **R** are mutually exclusive).

An example of the output of GT-PARA is shown in Figure 2. There is one predicate "take" in the sample input sentence. There are 4 arguments for that predicate, denoted as "302110", "AM-MOD", "203011", and "302012" respectively. "302110" symbolizes the NP Object of distance 1 prior to the passive predicate. "203011" symbolizes an undefined PP argument (which

Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL), pages 185–188, Ann Arbor, June 2005. ©2005 Association for Computational Linguistics

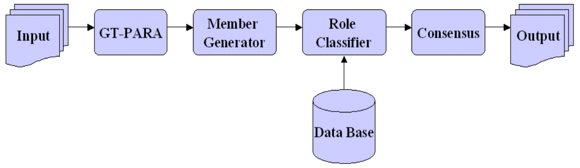

Figure 1: System Architecture

means it can be a core argument or an adjunct) with distance 1 after the passive predicate. And "302012" symbolizes a NP Object with distance 2 after the passive predicate.

For all boundaries extracted by GT-PARA, we simply denote all boundaries with noun phrases (NP) or similar phrases, such as WHNP, SBAR, and so on, as *core pattern candidates* and all boundaries with prepositional phrases (PP), ADJP, ADVP, or similar phrases, such as WHADJP, WHADVP, and so on, as *adjunct candidates*. But there is no exact rule for defining a core role or an adjunct explicitly in a boundary span, for example, given a sentence where

(1) P1 is done by P2. (P1 and P2 are two groups of words or phrases)

We can guess P1 might be labeled with "A1", and P2 with "A0" if there is no further feature information. But if the "head word" feature of P2 is "hour", for example, P2 can be labeled with "AM-TMP" instead. Because there are some uncertainties between core roles and adjuncts before labeling, we use the Member Generator (in Figure 1) to create all possible combinations, called members, from the output of GT-PARA by changing ANs (Core Role Candidates) into AMs (Adjunct Candidates), or AMs into ANs, except core candidates before predicates. All possible combinations (members) for the example in Figure 1 are

M1: [AN1, AM-MOD, V, AM1<points>(from), AN2] (original)
M2: [AN1 AM-MOD V *AN3* (from) AN2]
(change AM1 as AN3)
M3: [AN1 AM-MOD V AM1<point>(from) *AM2*<week>] (change AN2 as AM2)
M4: [AN1 AM-MOD V *AN3*<point>(from) *AM2*<week>]
(change AM1 as AN3 and one AN2 as AM2)

Words	POS	Full Tree Syntax	Predicate	Boundaries
The	DT	(S1(S(NP(NP*	-	(302110*
economy	NN	*	-	*
's	POS	*)	-	*
temperature	NN	*)	-	*)
will	MD	(VP*	-	(AM-MOD*)
be	AUX	(VP*	-	*
taken	VBN	(VP*	take	(V*V)
from	IN	(PP*	-	(203011*
several	JJ	(NP*	-	*
vantage	NN	*	-	*
points	NNS	*))	-	*)
this	DT	(NP*	-	(302012*
week	NN	*)	-	*)
.	.	*))	-	*

Figure 2: Illustration of an output of GT-PARA of a sentence, "The economy 's temperature will be taken from several vantage points this week."

The output from the Member Generator is passed to the Role Classifier, which finds all possible roles for each member with suitable core roles and adjuncts according to a Database built up by training data, in which each predicate has different patterns associated with it, each pattern has different semantic roles, and each role has the following format.

Role {Phrase type} < Head Word> (preposition)
There is an additional Boolean voice for a predicate to show if the predicate is passive or active (0: denotes active, 1: denotes passive). Each pattern includes a count on the number of the same patterns learned from the training data (denoted as "[*statistical figure*]"). For example, eight patterns for a predicate lemma "take" are

1. [30] A0{NP}<buyers> V{VP}<take>-0 A1{NP}<stake>
2. [1] A0{NP}<U.S.> V{VP}<take>-0 A1{NP}<%> A2{PP}<Canada>(from) AM-ADV{ADVP}<up>(up)
3. [2] A0{NP}<Confidence> V{VP}<take>-0 A1{NP}<dive> AM-ADV{SBAR}< figures>(if)
4. [1] A1{NP}<it> AM-MOD{VP}<could> V{VP}<take>-0 A2{NP}<place> AM-TMP{NP}<today> AM-LOC{PP}<Express>(at)
5. [1] AM-TMP{NP}< week> A0{NP}<government> V{VP}<take>-0 A1{NP}<bills> AM-DIR{PP}<to>(to)
6. [3] A1{NP}<cells> V{VP}<take>-1 A2{PP}<tissue>(from)
7. [6] A1{NP}<action> V{VP}<take>-1
8. [1] AM-TMP{ADVP}<far> A1{NP}<festivities> V{VP}<take>-1 AM-EXT{PP}<entirely> A0{NP}<eating>(by)

Role Classifier consists of two parts, AN classifier and AM classifier, which process core arguments and adjuncts respectively. AN classifier finds a suitable core pattern for labeled core pattern candidates in each member generated by Member Generator according to

(1) the same numbers of core roles
(2) the same **prepositions** for each core role
(3) the same **phrase types** for each core role
(4) the same voice (active or passive)

AM classifier finds a suitable adjunct role for any labeled adjunct candidate in each member generated by Member Generator according to

(1) the same Head Word
(2) the same Phrase type
(3) the highest statistical probability learned from the training data

The followings are the results for each member after Role Classification

M1: [AN1, AM-MOD, V, AM1<points>(from), AN2] (no pattern applied)

M2: [AN1 AM-MOD V AN1 (from) AN2] (no pattern applied)

M3: [*A1* AM-MOD V AM1<point>(from) *AM-TMP*<week>] (ANs by pattern 7, AM-TMP by pattern 5) [stat: 6]

M4: [*A1* AM-MOD V *A2* (from) *AM-TMP*<week>] (ANs by pattern 6, AM-TMP by pattern 5) [stat: 3]

Decision-making in the Consensus component (see Figure 1) handles the final selection by selecting the highest score using the following formula. $\text{Score}_k = (\alpha_1 * R_k + \alpha_2 * V_k + \alpha_3 * S_k)$ for each X_k (k=1 .. K, generated by Member Generator and Role Classifier), where

R_k : numbers of all roles being labeled
V_k : votes of a pattern with the same roles
S_k : statistical figure learned from trained data
X_k : different pattern by Member General and Role Classifier

α_1, α_2 ,and α_3 are weights ($\alpha_1 >> \alpha_2 >> \alpha_3$) used to rank the relative contribution of R_k , V_k , and S_k. Empirical studies led to the use of a so-called Max-labeled-role Heuristic to derive suitable values for these weights.

The final consensus decision for role classification is determined by calculating

$$\textbf{Consensus} = \operatorname*{Max}_{k=1}^{K} \text{Score}_k$$

There are 3 roles labeled in M3, which are AN1 as A1, AM-MOD, AM2 as AM-TMP respectively. And there are 4 roles labeled in M4, which are AN1 as A1, AM-MOD, AN3 as A2, and AM2 as AM-TMP respectively. Consensus scores for M3, and M4 are

$(\alpha_1 * 3 + \alpha_2 * 1 + \alpha_3 * 6)$, and
$(\alpha_1 * 4 + \alpha_2 * 1 + \alpha_3 * 3)$.

So the pattern [**A1** AM-MOD V **A2**(from) **AM-TMP**<week>] in M4 applied by Pattern 6 and Pattern 5 is selected due to the most roles labeled.

3 Data and Evaluation

We extracted patterns from the training data (WSJ Section 02 to 21) to build up a pattern database. Table 1 reveals sparseness of the pattern database. Twenty-six percent of predicates contain only one pattern, and fifteen two patterns. Seventy-five percents of predicates contain no more than 10 patterns.

No	1	2	3	4	5	5-10	11-50	51-100	>100
%	26	15	10	7	5	13	20	4	2
A %	26	40	50	57	62	75	94	98	100

Table 1: Statistical figures on the number of patterns collected from training, WSJ Section 02-21

The evaluation software, *srl-eval.pl*, is available from CoNLL2005 Shared Task[1] , which is the official script for evaluation of CoNLL-2005 Shared Task systems. In order to test boundary performance of GT-PARA, we simply convert all correct propositional arguments into A0s, except AM-MOD and AM-NEG for both the training dataset (WSJ Sections 15-18) and the development dataset (WSJ Section 24).

4 Experimental Results

The results of classification on the development, and test data of the CoNLL2005 shared task are outlined in Table 2. The overall results on the Development, Test-WSJ, Test-Brown, and Test-WSJ+Brown datasets for F-score are 65.78, 67.91, 58.58 and 66.72 respectively, which are moderate compared to the best result reported in CoNLL2004 Shared Task (Carreras et al., 2004) using partial trees and the result in (Pradhan et al., 2004). The results for boundary recognition via GT-PARA are summarized in Table 3.

[1] http://www.lsi.upc.edu/~srlconll/soft.html

	Precision	Recall	$F_{\beta=1}$
Development(WSJ24)	70.11%	61.96%	65.78
Test WSJ	71.49%	64.67%	67.91
Test Brown	65.75%	52.82%	58.58
Test WSJ + Brown	70.80%	63.09%	66.72

Test WSJ	Precision	Recall	$F_{\beta=1}$
Overall	71.49%	64.67%	67.91
A0	81.74%	81.53%	81.64
A1	71.61%	69.54%	70.56
A2	63.73%	40.36%	49.42
A3	68.60%	34.10%	45.56
A4	33.93%	18.63%	24.05
A5	0.00%	0.00%	0.00
AA	0.00%	0.00%	0.00
AM-ADV	36.26%	31.82%	33.89
AM-CAU	52.00%	35.62%	42.28
AM-DIR	20.11%	42.35%	27.27
AM-DIS	73.91%	63.75%	68.46
AM-EXT	12.90%	12.50%	12.70
AM-LOC	60.80%	33.33%	43.06
AM-MNR	43.57%	30.52%	35.90
AM-MOD	99.21%	90.93%	94.89
AM-NEG	96.38%	92.61%	94.46
AM-PNC	13.69%	31.30%	19.05
AM-PRD	0.00%	0.00%	0.00
AM-REC	0.00%	0.00%	0.00
AM-TMP	71.62%	54.55%	61.93
R-A0	93.37%	69.20%	79.49
R-A1	82.24%	56.41%	66.92
R-A2	100.00%	25.00%	40.00
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	0.00%	0.00%	0.00
R-AM-MNR	0.00%	0.00%	0.00
R-AM-TMP	0.00%	0.00%	0.00
V	97.34%	95.25%	96.29

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

The overall performance (F1: 76.43) on the WSJ Section 24 is not as good as on the WSJ Section 21 (F1: 85.78). The poor performance for the development was caused by more parser errors in the WSJ Section 24. Most parser errors are brought on by continuous phrases with commas and/or quotation marks.

One interesting fact is that when we tested our system using the data in CoNLL2004 shared task, we found the result with the train data WSJ 15-18 on the WSJ 21 is 73.48 shown in Table 4, which increases about 7 points in the F1 score, compared to WSJ 24 shown in Table 2. We found the labeling accuracy for WSJ 24 is 87.73, which is close to 89.30 for WSJ Section 21. But the results of boundary recognition in Table 3 for the two data are 9.14 points different, which leads to the better performance in WSJ Section 21. Boundary recognition as mentioned in CoNLL004 does play a very important role in this system as well.

	Precision	Recall	$F_{\beta=1}$
WSJ 15-18	87.23%	83.98%	85.57
WSJ 21	86.89%	84.70%	85.78
WSJ 24	78.88%	74.12%	76.43

Table 3: Boundary Recognition results by GT-PARA on WSJ 15-18, WSJ 21 and WSJ 24 sets

WSJ 21	Precision	Recall	$F_{\beta=1}$
Overall	78.06%	69.41%	73.48

Table 4: System results by the training data WSJ 15-18 on the WSJ Section 21

5 Conclusion

We have described a semantic role labeling architecture via consensus in a pattern-matching system. The pattern-matching system is based on linear pattern matching utilising statistical consensus for decision-making. A General Tree-based Predicate-Argument Boundary Recognition Algorithm (GT-PARA) handles the conversion process, turning a parse tree into a flat representation with all predicates and their arguments labeled with some useful features, such as phrase types. Label accuracy of Consensus model for role classification is stable but performance results of GT-PARA vary on different datasets, which is the key role for the overall results. Although the results seem moderate on test data, this system offers a decidedly different approach to the problem of semantic role labeling.

References

Xavier Carreras, Lluís Màrquez and Grzegorz Chrupała. 2004. Hierarchical Recognition of Propositional Arguments with Perceptrons. In *Proceeding of CoNLL'2004 Shared Task.*

Pradhan, S., Ward, W., Hacioglu, K., Martin, J., Jurafsky, D. 2004. " Shallow Semantic Parsing using Support Vector Machines ", in *Proceedings of HLT/NAACL-2004*, Boston, MA.