SENSEVAL-3: Third International Workshop on the Evaluation of Systems
for the Semantic Analysis of Text, Barcelona, Spain, July 2004
Association for Computational Linguistics

SVM Classification of FrameNet Semantic Roles

Dan Moldovan
Computer Science Dept.
University of Texas at Dallas
moldovan@utdallas.edu

Roxana Girju
Computer Science Dept.
Baylor University
girju@cs.baylor.edu

Marian Olteanu
and **Ovidiu Fortu**
Dept. of Computer Science
University of Texas at Dallas
marian@hlt.utdallas.edu
fovidiu@hlt.utdallas.edu

Abstract

A Support Vector Machines (SVM) classifier of FrameNet semantic roles was implemented based on a set of new and previously used syntactic and semantic features. At Senseval 3, the system achieved a precision of 0.807 for the restricted test and 0.898 for the non-restricted test.

1 Introduction

This paper reports on a system and results obtained for the Senseval 3 Semantic Roles task. The test data consists of 8002 sentences from 40 frames. The first task (non-restricted task) is to classify the semantic roles when their boundaries are predefined, and the second task (restricted task) is to perform argument boundary detection (ABD) and to classify the roles. Both the training and test data were divided into three groups: verb target, noun target and adjective target. Each FrameNet sentence was parsed with Charniak's parser. In our experiments we used the Support Vectors Machine (SVM) learning model.

2 System Description

The system architectures for the two tasks are presented in Figures 1 and 2.

2.1 The Feature Vector

We selected a list of 16 lexico-syntactic and semantic features split into three sets: the *baseline feature set*, the *modified feature set*, and the *new feature set*. As baseline we selected the list of eight features introduced and tested on the FrameNet corpus by (Gildea-Jurafsky, 2002). The modified feature set includes extensions we made to some already coined features, and the new feature set is the original contribution of this research. Table 1 shows all three sets of features along with their definitions; a detailed description is presented next.

Modified features

The **Governing Category feature**, as defined by Gildea & Jurafsky, works only for noun phrases and indicates the syntactic category of the phrase that governs the grammatical realization of the NP as the subject (eg, S), or the object (eg, VP) of the predicate. We extended this feature to *prepositional phrases* and *subordinate clauses* (eg, WH-clauses, SBAR).

The **Grammatical rule** associates the meaning of a verb with its syntactic behavior and expands the common ancestor of the argument and the target verb (eg, S $\Rightarrow$ NP+VP, for ARG0), as shown in Figure 3. This feature happens to include as a special case the *subcategorization* feature of (Pradhan et al. 2003) which expands only the predicate's parent node in the parse tree (eg, VP $\Rightarrow$ VBP+NP for ARG2).

New features

As defined by (Gildea-Jurafsky, 2002), and respectively by (Surdeanu et. al, 2003), the **Predicate** feature consists of two components: the verb lemma and the verb lexical item. We extended this set of components with a third one: the verb's part of speech.

The **Argument Structure** feature is similar to the Grammatical rule, but captures the syntactic phrase structure of the nodes along the path between the root node of the argument and its head word in a level-order fashion. For example, the argument structure of the argument ARG0 in Figure 3 is NP+NP*PP+JJ*JJ*NNS. The nodes that are part of the same subtree but not on the path to the head word are considered optional.

The **Frame** feature represents the frame in FrameNet that contains the target verb. The major disadvantage of this feature is that it is domain-dependent and it cannot be used but in correlation with FrameNet.

The **Distance** feature indicates the numeric distance between the argument and the target.

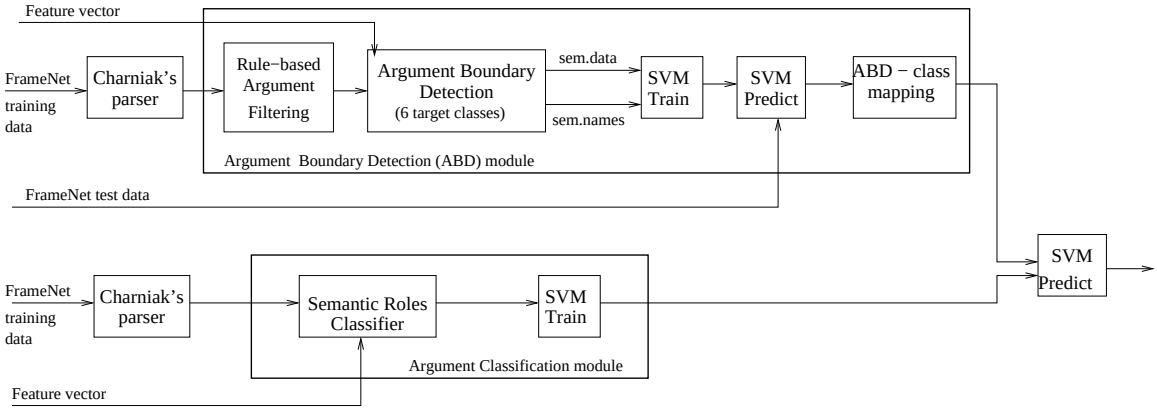

Figure 1: System architecture for the restricted run.

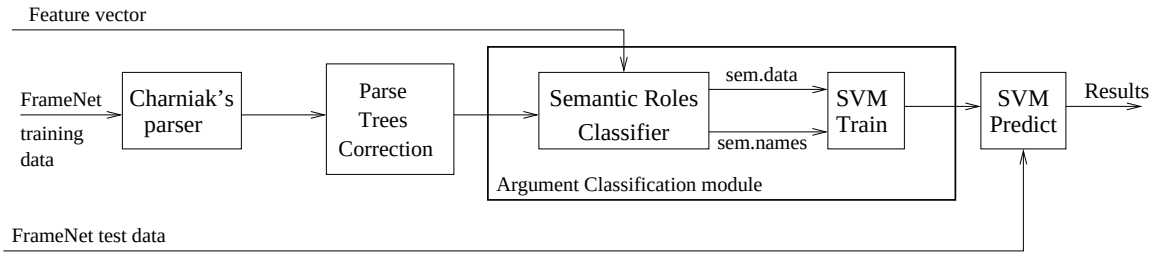

Figure 2: System architecture for the non-restricted run.

This is based on the observation that non-core elements are likely to be further from the target.

The **PropBank semantic argument** captures the semantic type of the argument. In order to obtain this feature we annotated FrameNet with the semantic arguments learned from PropBank using features F1 through F9. For example, the NP-SBJ phrase in Figure 3 was tagged as ARG0.

The **Diathesis alternation** feature represents a flat representation of the predicate argument structure in a sentence tagged with PropBank roles. For example, in Figure 3 the value of the feature is: ARG0 + V + ARG2.

2.2 Argument Boundary Detection

For argument boundary detection we used for training the subtrees of the 40-frames sentences corpus from which we eliminated the test set. Each subtree represents a training instance. Instead of using a binary ABD task (NULL vs. NON-NULL), we used six target classes: (1) non-argument, (2) perfect match, (3) potential argument, (4) subtree contains argument, (5) partial overlap (not inclusion), and (6) argument contains subtree. The classifier was trained on these six classes, and for the test data, the last four subclasses were taken into account as arguments.

2.3 Parse Trees Correction

For the non-restricted task where the arguments' boundaries were known, we improved the parse trees before the feature extraction phase. The procedure consists of joining two siblings into a single phrase node, or in moving away a subtree from a phrase node when necessary. Applying the two heuristics increased the coverage of arguments for verbs from 85% to about 97%.

2.4 SVM Learning Model

We applied the **Support Vector Machines** learning model to the semantic role classification problem and obtained encouraging results.

To discriminate among all the semantic roles classes, we use the "one vs. one" approach which constructs for each pair of classes a classifier which separates those classes.

The semantic roles classifier was trained on all the FrameNet data excluding the test set using the sixteen features. The multiclass classifier was then applied to the test data.

The software used in these experiments is the

No.	Feature	Definition
BASELINE FEATURES		
F1	**Phrase type** (G&J)	indicates the syntactic category of the argument to be tagged with the semantic role
F2	**Governing category** (G&J)	is provided only for noun phrases and indicates the syntactic category of the phrase that governs the grammatical realization of the NP as the subject (eg, S), or the object (eg, VP) of the predicate
F3	**Parse tree path** (G&J)	the path in the parse tree from the constituent argument to the predicate
F4	**Position** (G&J)	indicates whether the argument appears before or after the predicate
F5	**Voice** (G&J)	indicates if the verb is in active or passive voice in the sentence.
F6	**Head word** (G&J), (Surdeanu et al.)	indicates the syntactic head of the phrase; in our implementation, it coincides with the content word of (Surdeanu et al.) when applied to prepositional phrases and subordinate clauses;
F7	**Lemma** (G&J),	represents the lemma of the target word
F8	**Predicate** (Surdeanu et al.)	represents the surface form of the target word
MODIFIED FEATURES		
F9 (F2')	**Governing category**	we extended this feature to *prepositional phrases* (PP-prep.) and *subordinate clauses* (WH-, SBAR)
F10	**Grammatical rule**	the grammatical rule expanding the common ancestor of argument and verb; includes as special case the *subcategorization* feature of (Pradhan et al. 2003)
NEW FEATURES		
F11	**Predicate**	we added a third component: (C3) *verb's POS*
F12	**Argument structure**	the grammatical rules expanding all the nodes along the path to the head word
F13	**Frame**	FrameNet frame of the verb
F14	**Distance**	number of tokens between the argument and target
F15	**PropBank semantic argument**	the PropBank semantic type of the argument
F16	**Diathesis alternation**	same structure as grammatical rule except it uses the Propbank semantic argument feature instead of phrase type

Table 1: The three sets of features used for the automatic semantic role classification.

Runs	Results						
	Nr. correct	Nr. attempted	Nr. total	P	O	R	A
Non-restricted	13,660	15,208	16,279	0.898	0.897	0.839	93.4%
Restricted	12,697	15,735	16,279	0.807	0.777	0.780	96.7%

Table 2: Table of results for both runs. "P" is precision, "O" is overlap, "R" is recall, and "A" is attempted.

package LIBSVM, *http://www.csie.ntu.edu.tw/cjlin/libsvm/* which implements the SVM algorithm described above.

3 Experimental Setting and Results

3.1 Feature vectors

The data for SVM had to be transformed into a sparse vector format. The method used was to give each feature a number of slots in the vector equal to the number of its values, and put value 1 on the position that corresponds to the actual value of the feature for a data example and 0 for the rest. The range of values for each feature are listed in parentheses: phrase type (20), voice (2), headWord (16712) governingCategory (786) path (2091), position (2), lemma (1908), predicate (5235), posVerb (8), argumentStructure (1510), alternation (243), grammaticalFunction (8), and frame (266).

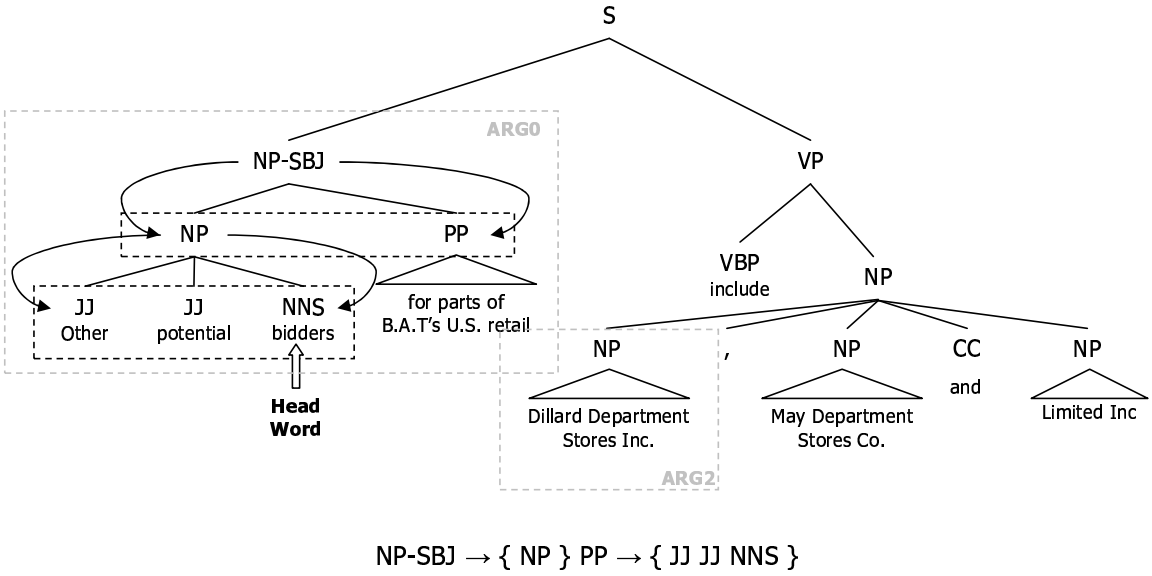

Figure 3: Example of Argument structure feature.

3.2 Model selection, training and testing

We used the RBF kernel $e^{-\gamma\|\mathbf{x}_i-\mathbf{x}_j\|}$ for our experiments. There is need to find the best values for γ and C, the cost coefficient of the optimization problem($min\frac{1}{2}\mathbf{w}^T\mathbf{w}+C\sum_{i=1}^{l}\xi_i$). This model selection is done by a grid search procedure. The estimation of γ and C takes from 10 to 20 hours on a single 3Gz Pentium 4 machine, while the final training on all data takes only a few hours.

3.3 Results

A summary of the results is shown in Table 2.

4 Acknowledgment

We acknowledge the contribution of A.M. Giuglea who worked on the feature extraction software.

References

C. Baker, C. Fillmore, and J. Lowe. The Berkeley FrameNet Project. In *Proceedings of COLLING/ACL*, Canada, 1998.

Daniel Gildea and Daniel Jurafsky. 2002. *Automatic Labeling of Semantic Roles*. In *Computational Linguistics*, 28(3), 2002.

P. Kingsbury, M. Palmer, and M. Marcus. Adding Semantic Annotation to the Penn TreeBank. In *Proceedings of the Human Language Technology Conference* (HLT 2002), California, 2002.

S. Pradhan, K. Hacioglu, V. Krugler, W. Ward, J. Martin, and D. Jurafsky. 2003. Semantic role parsing: Adding semantic structure to unstructured text. In *ICDM*, Florida.

M. Surdeanu et. al. 2003. Proceedings of the *Association for Computational Linguistics (ACL-2003)*, Sapporo, Japan, 2003.