Exploring Multilingual Semantic Role Labeling

Baoli Li, Martin Emms, Saturnino Luz, Carl Vogel
Department of Computer Science
Trinity College Dublin
Dublin 2, Ireland
{baoli.li,mtemms,luzs,vogel}@cs.tcd.ie

Abstract

This paper describes the multilingual semantic role labeling system of Computational Linguistics Group, Trinity College Dublin, for the CoNLL-2009 SRLonly closed shared task. The system consists of two cascaded components: one for disambiguating predicate word sense, and the other for identifying and classifying arguments. Supervised learning techniques are utilized in these two components. As each language has its unique characteristics, different parameters and strategies have to be taken for different languages, either for providing functions required by a language or for meeting the tight deadline. The system obtained labeled F1 69.26 averaging over seven languages (Catalan, Chinese, Czech, English, German, Japanese, and Spanish), which ranks the system fourth among the seven systems participating the SRLonly closed track.

1 Introduction

Semantic role labeling, which aims at computationally identifying and labeling arguments of predicate words, has become a leading research problem in computational linguistics with the advent of various supporting resources (e.g. corpora and lexicons) (Màrquez et al., 2008). Word semantic dependencies derived by semantic role labeling are assumed to facilitate automated interpretation of natural language texts. Moreover, techniques for automatic annotation of semantic dependencies can also play an important role in adding metadata to corpora for the purposes of machine translation and speech processing. We are currently investigating such techniques as part of our research into integrated language technology in the Center for Next Generation Localization (CNGL, http://www.cngl.ie). The multilingual nature of the CoNLL-2009 shared task on syntactic and semantic dependency analysis, which includes Catalan, Chinese, Czech, English, German, Japanese, and Spanish (Hajič et al., 2009), makes it a good testbed for our research.

We decided to participate in the CoNLL-2009 shared task at the beginning of March, signed the agreement for getting the training data on March 2nd, 2009, and obtained all the training data (especially the part from LDC) on March 4th, 2009. Due to the tight time constraints of the task, we chose to use existing packages to implement our system. These time constraints also meant that we had to resort to less computationally intensive methods to meet the deadline, especially for some large datasets (such as the Czech data). In spite of these difficulties and resource limitations, we are proud to be among the 21 teams who successfully submitted the results[1].

As a new participant, our goals in attending the CoNLL-2009 SRLonly shared task were to gain more thorough knowledge of this line of research and its state-of-the-art, and to explore how well a system quickly assembled with existing packages can fare at this hard semantic analysis problem.

Following the successful approaches taken by the participants of the CoNLL-2008 shared task (Surdeanu et al., 2008) on monolingual syntactic and semantic dependency analysis, we designed and implemented our CoNLL-2009 SRLonly system with pipeline architecture. Two main components are cascaded in this system: one is for disambiguating predicate word sense[2], and the other for identifying and classifying arguments for

[1] According to our correspondence with Dr. Jan Hajič, totally 31 teams among 60 registered ones signed and got the evaluation data.

[2] As predicate words are marked in the CoNLL-2009 datasets, we don't need to identify predicate words.

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 73–78,
Boulder, Colorado, June 2009.

predicate words. Different supervised learning techniques are utilized in these two components. For predicate word sense disambiguation (WSD), we have experimented with three algorithms: SVM, kNN, and Naïve Bayes. Based on experimental results on the development datasets, we chose SVM and kNN to produce our submitted official results. For argument identification and classification, we used a maximum entropy classifier for all the seven datasets. As each language has its unique characteristics and peculiarities within the dataset, different parameters and strategies have to be taken for different languages (as detailed below), either for providing functions required by a language or for meeting the tight deadline. Our official submission obtained 69.26 labeled F1 averaging over the seven languages, which ranks our system fourth among the seven systems in the SRLonly closed track.

The rest of this paper is organized as follows. Section 2 discusses the first component of our system for predicate word sense disambiguation. Section 3 explains how our system detects and classifies arguments with respect to a predicate word. We present experiments in Section 4, and conclude in Section 5.

2 Predicate Word Sense Disambiguation

This component tries to determine the sense of a predicate word in a specific context. As a sense of a predicate word is often associated with a unique set of possible semantic roles, this task is also called role set determination. Based on the characteristics of different languages, we take different strategies in this step, but the same feature set is used for different languages.

2.1 Methods

Intuitively, each predicate word should be treated individually according to the list of its possible senses. We therefore designed an initial solution based on the traditional methods in WSD: represent each sense as a vector from its definition or examples; describe the predicate word for disambiguation as a vector derived from its context; and finally output the sense which has the highest similarity with the current context. We also considered using singular value decomposition (SVD) to overcome the data sparseness problem. Unfortunately, we found this solution didn't work well in our preliminary experiments. The main problem is that the definition of each sense of a predicate word is not available. What we have is just a few example contexts for one sense of a predicate word, and these contexts are often not informative enough for WSD. On the other hand, our limited computing resources could not afford SVD operation on a huge matrix.

We finally decided to take each sense tag as a class tag across different words and transform the disambiguation problem into a normal multi-class categorization problem. For example, in the English datasets, all predicates with "01" as a sense identifier were counted as examples for the class "01". With this setting, a predicate word may be assigned an invalid sense tag. It is an indirect solution, but works well. We think there are at least two possible reasons: firstly, most predicate words take their popular sense in running text. For example, in the English dataset (training and development), 160,477 of 185,406 predicate occurrences (about 86.55%) take their default sense "01". Secondly, predicates may share some common role sets, even though their senses may not be exactly the same, e.g. "tell" and "inform".

Unlike the datasets in other languages, the Japanese dataset doesn't have specialized sense tags annotated for each predicate word, so we simply copy the predicted lemma of a predicate word to its PRED field. For other datasets, we derived a training sample for each predicate word, whose class tag is its sense tag. Then we trained a model from the generated training data with a supervised learning algorithm, and applied the learned model for predicting the sense of a predicate word. This is our base solution.

When transforming the datasets, the Czech data needs some special processing because of its unique annotation format. The sense annotation for a predicate word in the Czech data does not take the form "LEMMA.SENSE". In most cases, no specialized sense tags are annotated. The PRED field of these words only contains "LEMMA". In other cases, the disambiguated senses are annotated with an internal representation, which is given in a predicate word lexicon. We decomposed the internal representation of each predicate word into two parts: word index id and sense tag. For example, from "zvýšení v-w10004f2" we know "v-w10004" is the index id of word "zvýšení", and "f2" is its sense tag. We then use these derived

sense tags as class tags and add a class tag "=" for samples without specialized sense tag.

For each predicate word, we derive a vector describing its context and attributes, each dimension of which corresponds to a feature. We list the feature types in the next subsection. Features appearing only once are removed. The TF*IDF weighting schema is used to calculate the weight of a feature.

Three different algorithms were tried during the development period: support vector machines (SVM), distance-weighted k-Nearest Neighbor (kNN) (Li et al., 2004), and Naïve Bayes with multinomial model (Mccallum and Nigam, 1998). As to the SVM algorithm, we used the robust LIBSVM package (Chang and Lin, 2001), with a linear kernel and default values for other parameters. The algorithms achieving the best results in our preliminary experiments are chosen for different languages: SVM for Catalan, Chinese, and Spanish; kNN for German (k=20).

We used kNN for English (k=20) and Czech (k=10) because we could not finish training with SVM on these two datasets in limited time. Even with kNN algorithm, we still had trouble with the English and Czech datasets, because thousands of training samples make the prediction for the evaluation data unacceptably slow. We therefore had to further constrain the search space for a new predicate word to those samples containing the same predicate word. If there are not samples containing the same predicate word in the training data, we will assign it the most popular sense tag (e.g. "01" for English).

How to use the provided predicate lexicons is a challenging issue. Lexicons for different languages take different formats and the information included in different lexicons is quite different. We derived a sense list lexicon from the original predicate lexicon for Chinese, Czech, English, and German. Each entry in a sense list lexicon contains a predicate word, its internal representation (especially for Czech), and a list of sense tags that the predicate can have. Then we obtained a variant of our base solution, which uses the sense list of a predicate word to filter impossible senses. It works as follows:

- Disambiguate a new predicate with the base solution;
- Choose the most possible sense from all the candidate senses obtained in step 1: if the base classifier doesn't output a vector of probabilities for classes, only check whether the predicted one is a valid sense for the predicate;
- If there is not a valid sense for a new predicate (including the cases where the predicate does not have an entry in the sense list lexicon), output the most popular sense tag;

Unfortunately, preliminary experiments on the German and Chinese datasets didn't support to include such a post-processing stage. The performance with this filtering became a little worse. Therefore, we decided not to use it generally, but one exception is for the Czech data.

With kNN algorithm, we can greatly reduce the time for training the Czech data, but we do have problem with prediction, as there are totally 469,754 samples in the training dataset. It's a time-consuming task to calculate the similarities between a new sample and all the samples in the training dataset to find its k nearest neighbors, thus we have to limit the search space to those samples that contain the predicate word for disambiguation. To process unseen predicate words, we used the derived sense list lexicon: if a predicate word for disambiguation is out of the sense list lexicon, we simply copy its predicted lemma to the PRED field; if no sample in the training dataset has the same predicate word, we take its first possible sense in the sense list lexicon. With this strategy, our system can process the huge Czech dataset in short time.

2.2 Features

The features we used in this step include[3]:

a. [Lemma | (Lemma with POS)] of all words in the sentence;
b. Attributes of predicate word, which is obtained from PFEAT field by splitting the field at symbol "|" and removing the invalid attribute of "*";
c. [Lemma | POS] bi-grams of predicate word and its [previous | following] one word;
d. [Lemma | POS] tri-grams of predicate word and its [previous | following] two words;
e. [Lemma | (Lemma with POS)] of its most [left | right] child;
f. [(Lemma+Dependency_Relation+Lemma) | (POS +Dependency_Relation+POS)] of predicate word and its most [left | right] child;

[3] We referred to those CoNLL-2008 participants' reports, e.g. (Ciaramita et al., 2008), when we designed the feature sets for the two components.

g. [Lemma | (Lemma with POS)] of the head of the predicate word;
h. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of predicate word and its head;
i. [Lemma | (Lemma with POS)] of its [previous | following] two brothers;
j. [Lemma | POS | (Dependency relation)] bi-gram of predicate word and its [previous | following] one brother;
k. [Lemma | POS | (Dependency relation)] tri-gram of predicate word and its [previous | following] two brothers.

3 Argument Identification and Classification

The second component of our system is used to detect and classify arguments with respect to a predicate word. We take a joint solution rather than solve the problem in two consecutive steps: argument identification and argument classification.

3.1 Methods

By introducing an additional argument type tag "_" for non-arguments, we transformed the two tasks (i.e. argument identification and argument classification) into one multi-class classification problem. As a word can play different roles with respect to different predicate words and a predicate word can be an argument of itself, we generate a training set by deriving a training example from each word-predicate pair. For example, if a sentence with two predicates has 7 words, we will derive 7*2=14 training examples. Therefore, the number of training examples generated in this step will be around L times larger than that obtained in the previous step, where L is the average length of sentences.

We chose to use maximum entropy algorithm in this step because of its success in the CoNLL-2008 shared task (Surdeanu et al., 2008). Le Zhang's maximum entropy package (Zhang, 2006) is integrated in our system.

The Czech data cause much trouble again for us, as the training data derived by the above strategy became even larger. We had to use a special strategy for the Czech data: we selectively chose word-predicate pairs for generating the training dataset. In other words, not all possible combinations are used. We chose the following words with respect to each predicate: the first and the last two words of a sentence; the words between the predicate and any argument of it; two words before the predicate or any argument; and two words after the predicate or any argument.

In the Czech and Japanese data, some words may play multiple roles with respect to a predicate word. We thus have to consider multi-label classification problem (Tsoumakas and Katakis, 2007) for these two languages' data. We tried the following two solutions:

- Take each role type combination as a class and transform the multi-label problem to a single-label classification problem;
- Classify a word with a set of binary classifiers: consider each role type individually with a binary classifier; any possible role type will be output; if no role type is obtained after considering all the role types, the role type with the highest confidence value will be output; and, if "_" is output with any other role type, remove it.

We used the second solution in our official submission, but we finally found these two solutions perform almost the same. The performance difference is very small. We found the cases with multi-labels (actually at most two) in the training data are very limited: 690 of 414,326 in the Czech data and 113 of 46,663 in the Japanese data.

3.2 Features

The features we used in this step include:

a. Whether the current word is a predicate;
b. [Lemma | POS] of current word and its [previous | following] one word;
c. [Lemma | POS] bi-grams of current word and its [previous | following] one word;
d. POS tri-grams of current word, its previous word and its following word;
e. Dependency relation of current word to its head;
f. [Lemma | POS] of the head of current word;
g. [Lemma | POS] bi-grams of current word and its head;
h. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of current word and its head;
i. [Lemma | POS] of its most [left | right] child;
j. [Lemma | POS] bi-grams of current word and its most [left | right] child;
k. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS) of current word and its most [left | right] child;
l. The number of children of the current word and the predicate word;
m. Attributes of the current word, which is obtained from PFEAT field by splitting the field at symbol "|" and removing the invalid attribute of "*";
n. The sense tag of the predicate word;

o. [Lemma | POS] of the predicate word and its head;
p. Dependency relation of the predicate word to its head;
q. [Lemma | POS] bi-grams of the predicate word and its head;
r. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of the predicate word and its head;
s. [Lemma | POS] of the most [left | right] child of the predicate word;
t. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of predicate word and its head;
u. [Lemma | POS] bi-gram of the predicate word and its most [left | right] child;
v. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of the predicate word and its most [left | right] child;
w. The relative position of the current word to the predicate one: before, after, or on;
x. The distance of the current word to the predicate one;
y. The relative level (up, down, or same) and level difference on the syntactic dependency tree of the current word to the predicate one;
z. The length of the shortest path between the current word and the predicate word.

4 Experiments

4.1 Datasets

The datasets of the CoNLL-2009 shared task contain seven languages: Catalan (CA), Chinese (CN), Czech (CZ), English (EG), German (GE), Japanese (JP), and Spanish (SP). The training and evaluation data of each language (Taulé et al., 2008; Xue et al., 2008; Hajič et al., 2006; Palmer et al., 2002; Burchardt et al., 2006; Kawahara et al., 2002) have been converted to a uniform CoNLL Shared Task format. Each participating team is required to process all seven language datasets.

Lanuage	CA	CN	CZ	EN	GE	JP	SP
Size (KB)	48974	41340	**94284**	58155	41091	8948	52430
# of Sentences	14924	24039	**43955**	40613	38020	4643	15984
# of Predicate words	42536	110916	**469754**	185404	17988	27251	48900
Avg. # of Predicates per sentence	2.85	4.61	**10.69**	4.57	0.47	5.87	3.06
popular sense tag	a2 (37%)	01 (90%)	= (81%)	01 (87%)	1 (75%)	= (100%)	a2 (39%)

Table 1. Statistical information of the seven language datasets (training and development).

Table 1 shows some statistical information of both training and development data for each language. The total size of the uncompressed original data without lexicons is about 345MB. The Czech dataset is the largest one containing 43,955 sentences and 469,754 predicate words, while the Japanese dataset the smallest one. On average, 10.69 predicate words appear in a Czech sentence, while only 0.47 predicate words exist in a German sentence. The most popular sense tag in the Czech datasets is "=", which means the PRED field has the same value as the PLEMMA field or the FORM field. About 81% of Czech predicate words take this value.

4.2 Experimental Results

F1 is used as the main evaluation metric in the CoNLL-2009 shared task. As to the SRLonly track, a joint semantic labeled F1, which considers predicate word sense disambiguation and argument labeling equally, is used to rank systems.

Avg.	CA	CN	CZ	EG	GE	JP	SP
69.26	**74.06**	70.37	57.46	69.63	67.76	72.03	73.54

Table 2. Official results of our system.

Table 2 gives the official results of our system on the evaluation data. The system obtained the best result (74.06) on the Catalan data, but performed very poor (57.46) on the Czech data. Except the Czech data, our system performs quite stable on the other six language data with mean of 71.23 and standard deviation of 2.42.

	Avg.	CA	CN	CZ	EG	GE	JP	SP
Overall F1	**69.47**	**74.12**	**70.52**	**57.57**	**70.24**	**67.97**	**72.17**	**73.68**
Pred. WSD F1	**86.9**	**84.42**	**94.54**	**72.23**	**92.98**	**81.09**	**99.07**	**83.96**
Arg I&C F1	**57.24**	**69.29**	**57.71**	**33.19**	**58.25**	**60.64**	**52.72**	**68.86**
Arg I&C PR	69.77	73.43	72.48	62.14	70.14	66.63	69.37	74.23
Arg I&C RE	49.77	65.6	47.94	22.64	49.81	55.64	42.52	64.21

Table 3. Results of our system after fixing a minor bug.

After submitting the official results, we found and fixed a minor bug in the implementation of the second component. Table 3 presents the results of our system after fixing this bug. The overall performance doesn't change much. We further analyzed the bottlenecks by checking the performance of different components.

At the predicate WSD part, our system works reasonable with labeled F1 86.9, but the performance on the Czech data is lower than that of a baseline system that constantly chooses the most popular sense tag. If we use this baseline solution,

we can get predicate WSD F1 78.66, which further increases the overall labeled F1 on the Czech data to 61.68 from 57.57 and the overall labeled F1 over the seven languages to 70.05 from 69.47.

From table 3, we can see our system performs relatively poorly for argument identification and classification (57.24 vs. 86.9). The system seems too conservative for argument identification, which makes the recall very lower. We explored some strategies for improving the performance of the second component, e.g. separating argument identification and argument classification, and using feature selection (with DF threshold) techniques, but none of them helps much. We are thinking the features currently used may not be effective enough, which deserves further study.

5 Conclusion and Future Work

In this paper, we describe our system for the CoNLL-2009 shared task -- SRLonly closed track. Our system was built on existing packages with a pipeline architecture, which integrated two cascaded components: predicate word sense disambiguation and argument identification and classification. Our system performs well at disambiguating the sense of predicate words, but poorly at identifying and classifying arguments. In the future, we plan to explore much effective features for argument identification and classification.

Acknowledgments

This research was funded by Science Foundation Ireland under the CNGL grant. We used the IITAC Cluster in our initial experiments. We thank IITAC, the HEA, the National Development Plan and the Trinity Centre for High Performance Computing for their support. We are also obliged to John Keeney for helping us running our system on the CNGL servers.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó and Manfred Pinkal. 2006. The SALSA Corpus: a German Corpus Resource for Lexical Semantics. *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006).* Genoa, Italy.

Chih-Chung Chang and Chih-Jen Lin. 2001. LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Massimiliano Ciaramita, Giuseppe Attardi, Felice Dell'Orletta, and Mihai Surdeanu. 2008. DeSRL: A Linear-Time Semantic Role Labeling System. *Proceedings of the CoNLL-2008.*

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antonia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue and Yi Zhang. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009).* Boulder, Colorado, USA.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová and Zdeněk Žabokrtský. 2006. *The Prague Dependency Treebank 2.0.* Linguistic Data Consortium, USA. ISBN 1-58563-370-4.

Daisuke Kawahara, Sadao Kurohashi and Koiti Hasida. 2002. Construction of a Japanese Relevance-tagged Corpus. *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002).* Las Palmas, Spain.

Baoli Li, Qin Lu and Shiwen Yu. 2004. An Adaptive k-Nearest Neighbor Text Categorization Strategy. *ACM Transactions on Asian Language Information Processing*, 3(4): 215-226.

Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski and Suzanne Stevenson. 2008. Semantic Role Labeling: An Introduction to the Special Issue. *Computational Linguistics*, 34(2):145-159.

Andrew Mccallum and Kamal Nigam. 1998. A Comparison of Event Models for Naive Bayes Text Classification. *Proceedings of AAAI/ICML-98 Workshop on Learning for Text Categorization*.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluis Marquez and Joakim Nivre. 2008. The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL-2008).*

Mariona Taulé, Maria Antònia Martí and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008).* Marrakech, Morocco.

Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-Label Classification: An Overview. *International Journal of Data Warehousing and Mining*, 3(3):1-13.

Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering,* 15(1):143-172.

Le Zhang. 2006. Maximum Entropy Modeling Toolkit for Python and C++. Software available at http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html.