Multilingual Semantic Role Labeling

Anders Björkelund **Love Hafdell** **Pierre Nugues**
Department of Computer Science, Lund University
S-221 00 Lund, Sweden
fte04abj@student.lth.se
love_hafdell@hotmail.com
Pierre.Nugues@cs.lth.se

Abstract

This paper describes our contribution to the semantic role labeling task (SRL-only) of the CoNLL-2009 shared task in the closed challenge (Hajič et al., 2009). Our system consists of a pipeline of independent, local classifiers that identify the predicate sense, the arguments of the predicates, and the argument labels. Using these local models, we carried out a beam search to generate a pool of candidates. We then reranked the candidates using a joint learning approach that combines the local models and proposition features.

To address the multilingual nature of the data, we implemented a feature selection procedure that systematically explored the feature space, yielding significant gains over a standard set of features. Our system achieved the second best semantic score overall with an average labeled semantic F1 of 80.31. It obtained the best F1 score on the Chinese and German data and the second best one on English.

1 Introduction

In this paper, we describe a three-stage analysis approach that uses the output of a dependency parser and identifies the arguments of the predicates in a sentence. The first stage consists of a pipeline of independent classifiers. We carried out the predicate disambiguation with a set of greedy classifiers, where we applied one classifier per predicate lemma. We then used a beam search to identify the arguments of each predicate and to label them, yielding a pool of candidate propositions. The second stage consists of a reranker that we applied to the candidates using the local models and proposition features. We combined the score of the greedy classifiers and the reranker in a third stage to select the best candidate proposition. Figure 1 shows the system architecture.

We evaluated our semantic parser on a set of seven languages provided by the organizers of the CoNLL-2009 shared task: Catalan and Spanish (Taulé et al., 2008), Chinese (Palmer and Xue, 2009), Czech (Hajič et al., 2006), English (Surdeanu et al., 2008), German (Burchardt et al., 2006), and Japanese (Kawahara et al., 2002). Our system achieved an average labeled semantic F1 of 80.31, which corresponded to the second best semantic score overall. After the official evaluation was completed, we discovered a fault in the training procedure of the reranker for Spanish. The revised average labeled semantic F1 after correction was 80.80.

2 SRL Pipeline

The pipeline of classifiers consists of a predicate disambiguation (PD) module, an argument identification module (AI), and an argument classification (AC) module. Aside from the lack of a predicate identification module, which was not needed, as predicates were given, this architecture is identical to the one adopted by recent systems (Surdeanu et al., 2008), as well as the general approach within the field (Gildea and Jurafsky, 2002; Toutanova et al., 2005).

We build all the classifiers using the L2-regularized linear logistic regression from the LIBLINEAR package (Fan et al., 2008). The package implementation makes models very fast to train and

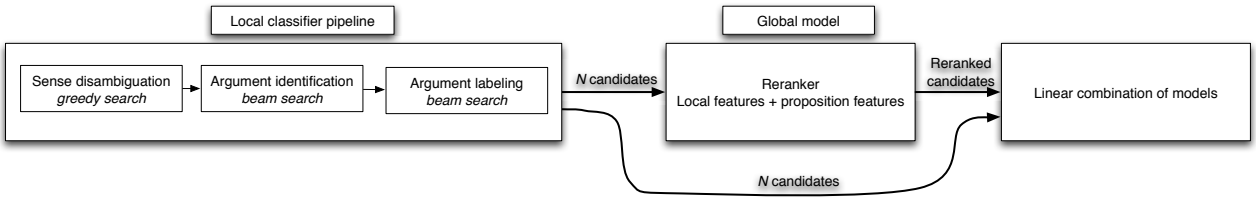

Figure 1: System architecture.

use for classification. Since models are logistic, they produce an output in the form of probabilities that we use later in the reranker (see Sect. 3).

2.1 Predicate Disambiguation

We carried out a disambiguation for all the lemmas that had multiple senses in the corpora and we trained one classifier per lemma. We did not use the predicate lexicons and we considered lemmas with a unique observed sense as unambiguous.

English required a special processing as the sense nomenclature overlapped between certain nominal and verbal predicates. For instance, the nominal predicate *plan.01* and the verbal predicate *plan.01* do not correspond to the same semantic frame. Hence, we trained two classifiers for each lemma *plan* that could be both a nominal and verbal predicate.

Table 1: Feature sets for predicate disambiguation.

	ca	**ch**	**cz**	**en**	**ge**	**sp**
PredWord	•		•			•
PredPOS	•			•		
PredDeprel		•	•	•		
PredFeats			•		•	•
PredParentWord	•	•	•	•	•	
PredParentPOS		•		•	•	
PredParentFeats			•		•	
DepSubCat	•	•	•	•	•	
ChildDepSet	•	•	•	•	•	•
ChildWordSet	•	•	•	•	•	•
ChildPOSSet		•	•	•	•	•

2.2 Argument Identification and Classification

We implemented the argument identification and classification as two separate stages, because it enabled us to apply and optimize different feature sets in each step. Arguments were identified by means of a binary classifier. No pruning was done, each word in the sentence was considered as a potential argument to all predicates of the same sentence.

Arguments were then labeled using a multiclass classifier; each class corresponding to a certain label. We did not apply any special processing with multiple dependencies in Czech and Japanese. Instead, we concatenated the composite labels (i.e. double edge) to form unique labels (i.e. single edge) having their own class.

2.3 Identification and Classification Features

For the English corpus, we used two sets of features for the nominal and the verbal predicates both in the AI and AC steps. This allowed us to create different classifiers for different kinds of predicates. We extended this approach with a default classifier catching predicates that were wrongly tagged by the POS tagger. For both steps, we used the union of the two feature sets for this catch-all class.

We wanted to employ this procedure with the two other languages, Czech and Japanese, where predicates had more than one POS type. As feature selection (See Sect. 2.4) took longer than expected, particularly in Czech due to the size of the corpus and the annotation, we had to abandon this idea and we trained a single classifier for all POS tags in the AI and AC steps.

For each data set, we extracted sets of features similar to the ones described by Johansson and Nugues (2008). We used a total of 32 features that we denote with the prefixes: Pred-, PredParent-, Arg-, Left-, Right-, LeftSibling-, and RightSibling- for, respectively, the predicate, the parent of the predicate, the argument, the leftmost and rightmost dependents of the argument, and the left and right

Table 2: Feature sets for argument identification and classification.

	Argument identification							Argument classification						
	ca	**ch**	**cz**	**en**	**ge**	**ja**	**sp**	**ca**	**ch**	**cz**	**en**	**ge**	**ja**	**sp**
PredWord									•		N		•	
PredPOS				N	•				•		V		•	
PredLemma				N	•			•	•	•	N,V	•		•
PredDeprel														
Sense		•	•	V	•			•	•	•	N,V	•	•	•
PredFeats			•							•		•		
PredParentWord				V					•		V		•	
PredParentPOS				V							V	•		
PredParentFeats			•											
DepSubCat	•	•												
ChildDepSet	•				•			•	•		V	•	•	•
ChildWordSet				N						•			•	
ChildPOSSet		•							•		N		•	
ArgWord	•	•		N,V	•	•	•	•	•	•	N,V	•	•	•
ArgPOS	•	•		N,V	•	•	•	•	•	•	N,V		•	
ArgFeats	•							•		•		•		•
ArgDeprel	•	•	•	V	•		•	•	•	•	V	•		•
DeprelPath	•	•	•	N,V	•	•	•	•	•		V	•		
POSPath	•	•	•	N,V	•	•	•			•	V	•	•	
Position			•	N,V		•		•	•	•	N,V		•	•
LeftWord	•				•			•	•		N		•	•
LeftPOS						•			•		V			
LeftFeats						•	•					•		
RightWord	•			N				•	•		N,V			•
RightPOS				N				•	•		N,V			•
RightFeats								•						•
LeftSiblingWord	•						•	•	•		N		•	
LeftSiblingPOS					•	•		•	•		N,V		•	
LeftSiblingFeats			•					•				•		
RightSiblingWord	•	•		V	•	•	•		•			•		•
RightSiblingPOS												•	•	
RightSiblingFeats													•	

sibling of the argument. The suffix of these names corresponds to the column name of the CoNLL format, except Word which corresponds to the Form column. Additional features are:

- Sense: the value of the Pred column, e.g. *plan.01*.
- Position: the position of the argument with respect to the predicate, i.e. before, on, or after.
- DepSubCat: the subcategorization frame of the predicate, e.g. OBJ+OPRD+SUB.
- DeprelPath: the path from predicate to argument concatenating dependency labels with the direction of the edge, e.g. OBJ↑OPRD↓SUB↓.
- POSPath: same as DeprelPath, but dependency labels are exchanged for POS tags, e.g. NN↑NNS↓NNP↓.
- ChildDepSet: the set of dependency labels of the children of the predicate, e.g. {OBJ, SUB}.
- ChildPOSSet: the set of POS tags of the children of the predicate, e.g. {NN, NNS}.
- ChildWordSet: the set of words (Form) of the children of the predicate, e.g. {*fish*, *me*}.

2.4 Feature Selection

We selected the feature sets using a greedy forward procedure. We first built a set of single features and, to improve the separability of our linear classifiers, we paired features to build bigrams. We searched the space of feature bigrams using the same procedure. See Johansson (2008, page 83), for a complete description. We intended to carry out a cross-validation search. Due to the lack of time, we resorted to using 80% of the training set for training and 20% for evaluating the features. Table 2 contains the complete list of single features we used. We omitted the feature bigrams.

Feature selection turned out to be a massive task. It took us three to four weeks searching the feature spaces, yet in most cases we were forced to interrupt the selection process after a few bigram features in order to have our system ready in time. This means that our feature sets can probably be further optimized.

When the training data was initially released, we used the exact feature set from Johansson and Nugues (2008) to compute baseline results on the development set for all the languages. After feature selection, we observed an increase in labeled semantic F1 close to 10% in most languages.

2.5 Applying Beam Search

The AI module proceeds left to right considering each word as an argument of the current predicate. The current partial propositions are scored by computing the product of the probabilities of all the words considered so far. After each word, the current pool of partial candidates is reduced to the beam size, k, and at the end of the sentence, the top k scoring propositions are passed on to the AC module.

Given k unlabeled propositions, the AC module applies a beam search on each of these propositions independently. This is done in a similar manner, proceeding from left to right among the identified arguments, keeping the l best labelings in its beam, and returning the top l propositions, when all identified arguments have been processed. This yields $n = k \times l$ complete propositions, unless one of the unlabeled propositions has zero arguments, in which case we have $n = (k - 1) \times l + 1$.

The probability of a labeled proposition according to the local pipeline is given by $P_{Local} = P_{AI} \times P_{AC}$, where P_{AI} and P_{AC} is the output probability from the AI and AC modules, respectively. In the case of empty propositions, P_{AC} was set to 1.

3 Global Reranker

We implemented a global reranker following Toutanova et al. (2005). To generate training examples for the reranker, we trained m AI and AC classifiers by partitioning the training set in m parts and using $m - 1$ of these parts for each AI and AC classifier, respectively.

We applied these AI and AC classifiers on the part of the corpus they were *not* trained on and we then generated the top n propositions for each predicate. We ran the CoNLL evaluation script on the propositions and we marked the top scoring one(s) as positive. We marked the others negative. If the correct proposition was not in the pool of candidates, we added it as an *extra* positive example. We used these positive and negative examples as training data for the global reranker.

3.1 Reranker Features

We used all the features from the local pipeline for all the languages. We built a vector where the AI features were prefixed with AI- and the AC features prefixed with $lab-$, where lab was any of the argument labels.

We added one proposition feature to the concatenation of local features, namely the sequence of core argument labels, e.g. *A0+plan.01+A1*. In Catalan and Spanish, we considered all the labels prefixed by arg0, arg1, arg2, or arg3 as core labels. In Chinese and English, we considered only the labels A0, A1, A2, A3, and A4. In Czech, German, and Japanese, we considered all the labels as core labels.

Hence, the total size of the reranker vector space is $|AI| + |L| \times |AC| + |G|$, where $|AI|$ and $|AC|$ denotes the size of the AI and AC vector spaces, respectively, $|L|$ corresponds to the number of labels, and $|G|$ is the size of additional global features.

We ran experiments with the grammatical voice that we included in the string representing the sequence of core argument labels, e.g. *A1+plan.01/Passive+A0*. The voice was derived by hand-crafted rules in Catalan, English, German, and

Spanish, and given in the Feat column in Czech. However, we did not notice any significant gain in performance. The hand-crafted rules use lexical forms and dependencies, which we believe classifiers are able to derive themselves using the local model features. This also applies to Czech, as PredFeats was a feature used in the local pipeline, both in the AI and AC steps.

3.2 Weighting the Models

In Sect. 2.5, we described how the pipeline was used to generate the top n propositions, each with its own local probability P_{Local}. Similar to softmax, we normalized these local probabilities by dividing each of them by their total sum. We denote this normalized probability by P'_{Local}. The reranker gives a probability on the complete proposition, $P_{Reranker}$. We weighted these probabilities and chose the proposition maximizing $P_{Final} = (P'_{Local})^{\alpha} \times P_{Reranker}$. This is equivalent to a linear combination of the log probabilities.

3.3 Parameters Used

For the submission to the CoNLL 2009 Shared Task, we set the beam widths to $k = l = 4$, yielding candidate pools of size $n = 13$ or $n = 16$ (See Section 2.5). We used $m = 5$ for training the reranker and $\alpha = 1$ for combining the local model with the reranker.

4 Results

Our system achieved the second best semantic score, all tasks, with an average labeled semantic F1 of 80.31. It obtained the best F1 score on the Chinese and German data and the second best on English. Our system also reached the third rank in the out-of-domain data, all tasks, with a labeled semantic F1 of 74.38. Post-evaluation, we discovered a bug in the Spanish reranker model causing the poor results in this language. After correcting this, we could reach a labeled semantic F1 of 79.91 in Spanish. Table 3 shows our official results in the shared task as well as the post-evaluation update.

We also compared the performance of a greedy strategy with that of a global model. Table 4 shows these figures with post-evaluation figures in Spanish. Table 5 shows the training time, parsing time, and the parsing speed in predicates per second. These figures correspond to complete execution time of parsing, including loading models into memory, i.e. a constant overhead, that explains the low parsing speed in German. We implemented our system to be flexible for easy debugging and testing various ideas. Optimizing the implementation would reduce execution times significantly.

Table 3: Summary of submitted results: closed challenge, semantic F1. * denotes the post-evaluation results obtained for Spanish after a bug fix.

	Unlabeled	**Labeled**
Catalan	93.60	80.01
Chinese	84.76	78.60
Czech	92.63	85.41
English	91.17	85.63
German	92.13	79.71
Japanese	83.45	76.30
Spanish	92.69	76.52
Spanish*	93.76	79.91
Average	90.06	80.31
Average*	90.21	80.80

Table 4: Improvement of reranker. * denotes the post-evaluation results obtained for Spanish after a bug fix.

	Greedy	**Reranker**	**Gain**
Catalan	79.54	80.01	0.47
Chinese	77.84	78.60	0.76
Czech	84.99	85.41	0.42
English	84.44	85.63	1.19
German	79.01	79.71	0.70
Japanese	75.61	76.30	0.69
Spanish	79.28	76.52	-2.76
Spanish*	79.28	79.91	0.63
Average	80.10	80.31	0.21
Average*	80.10	80.80	0.70

5 Conclusion

We have built and described a streamlined and effective semantic role labeler that did not use any lexicons or complex linguistic features. We used a generic feature selection procedure that keeps language adaptation minimal and delivers a relatively even performance across the data sets. The system is

Table 5: Summary of training and parsing times on an Apple Mac Pro, 3.2 GHz.

	Training (min)	**Parsing (Greedy)** (min:sec)	**Speed (Greedy)** (pred/sec)	**Parsing (Reranker)** (min:sec)	**Speed (Reranker)** (pred/sec)
Catalan	46	1:10	71	1:21	62
Chinese	139	2:35	79	3:45	55
Czech	299	18:47	40	33:49	22
English	421	6:25	27	8:51	20
German	15	0:21	26	0:22	25
Japanese	48	0:37	84	1:02	50
Spanish	51	1:15	69	1:47	48

robust and can handle incorrect syntactic parse trees with a good level of immunity. While input parse trees in Chinese and German had a labeled syntactic accuracy of 78.46 (Hajič et al., 2009), we could reach a labeled semantic F1 of 78.60 and 79.71 in these languages. We also implemented an efficient global reranker in all languages yielding a 0.7 average increase in labeled semantic F1. The reranker step, however, comes at the expense of parsing times increased by factors ranging from 1.04 to 1.82.

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. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia 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. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of the Shared Task Session of CoNLL-2008*.

Richard Johansson. 2008. *Dependency-based Semantic Analysis of Natural-language Text*. Ph.D. thesis, Lund University, December 5.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

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

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *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. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of ACL-2005*.