Mixing and Blending Syntactic and Semantic Dependencies

Yvonne Samuelsson
Dept. of Linguistics
Stockholm University
yvonne.samuelsson@ling.su.se

Oscar Täckström
Dept. of Linguistics and Philology
SICS / Uppsala University
oscar@sics.se

Sumithra Velupillai
Dept. of Computer and Systems Sciences
Stockholm University / KTH
sumithra@dsv.su.se

Johan Eklund
SSLIS
University College of Borås
johan.eklund@hb.se

Mark Fišel
Dept. of Computer Science
University of Tartu
fishel@ut.ee

Markus Saers
Dept. of Linguistics and Philology
Uppsala University
markus.saers@lingfil.uu.se

Abstract

Our system for the CoNLL 2008 shared task uses a set of individual parsers, a set of stand-alone semantic role labellers, and a joint system for parsing and semantic role labelling, all blended together. The system achieved a macro averaged labelled F_1-score of 79.79 (WSJ 80.92, Brown 70.49) for the overall task. The labelled attachment score for syntactic dependencies was 86.63 (WSJ 87.36, Brown 80.77) and the labelled F_1-score for semantic dependencies was 72.94 (WSJ 74.47, Brown 60.18).

1 Introduction

This paper presents a system for the CoNLL 2008 shared task on joint learning of syntactic and semantic dependencies (Surdeanu et al., 2008), combining a two-step pipelined approach with a joint approach.

In the pipelined system, eight different syntactic parses were blended, yielding the input for two variants of a semantic role labelling (SRL) system. Furthermore, one of the syntactic parses was used with an early version of the SRL system, to provide predicate predictions for a joint syntactic and semantic parser. For the final submission, all nine syntactic parses and all three semantic parses were blended.

The system is outlined in Figure 1; the dashed arrow indicates the potential for using the predicate prediction to improve the joint syntactic and semantic system.

© 2008. Licensed under the *Creative Commons Attribution-Noncommercial-Share Alike 3.0 Unported* license (http://creativecommons.org/licenses/by-nc-sa/3.0/). Some rights reserved.

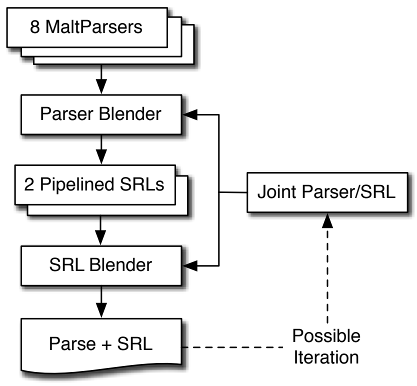

Figure 1: Overview of the submitted system.

2 Dependency Parsing

The initial parsing system was created using MaltParser (Nivre et al., 2007) by blending eight different parsers. To further advance the syntactic accuracy, we added the syntactic structure predicted by a joint system for syntactic and semantic dependencies (see Section 3.4) in the blending process.

2.1 Parsers

The MaltParser is a dependency parser generator, with three parsing algorithms: Nivre's arc standard, Nivre's arc eager (see Nivre (2004) for a comparison between the two Nivre algorithms), and Covington's (Covington, 2001). Both of Nivre's algorithms assume projectivity, but the MaltParser supports pseudo-projective parsing (Nilsson et al., 2007), for projectivization and deprojectivization.

	WSJ	Brown
Best single parse	85.22%	78.37%
LAS weights	87.00%	80.60%
Learned weights	87.36%	80.77%

Table 1: Labelled attachment score on the two test sets of the best single parse, blended with weights set to PoS labelled attachment score (LAS) and blended with learned weights.

Four parsing algorithms (the two Nivre algorithms, and Covington's projective and non-projective version) were used, creating eight parsers by varying the parsing direction, left-to-right and right-to-left. The latter was achieved by reversing the word order in a pre-processing step and then restoring it in post-processing. For the final system, feature models and training parameters were adapted from Hall et al. (2007).

2.2 Blender

The single parses were blended following the procedure of Hall et al. (2007). The parses of each sentence were combined into a weighted directed graph. The Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) was then used to find the maximum spanning tree (MST) of the graph, which was considered the final parse of the sentence. The weight of each graph edge was calculated as the sum of the weights of the corresponding edges in each single parse tree.

We used a simple iterative weight updating algorithm to learn the individual weights of each single parser output and part-of-speech (PoS) using the development set. To construct an initial MST, the labelled attachment score was used. Each single weight, corresponding to an edge of the hypothesis tree, was then iteratively updated by slightly increasing or decreasing the weight, depending on whether it belonged to a correct or incorrect edge as compared to the reference tree.

2.3 Results

The results are summarized in Table 1; the parse with LAS weights and the best single parse (Nivre's arc eager algorithm with left-to-right parsing direction) are also included for comparison.

3 Semantic Role Labelling

The SRL system is a pipeline with three chained stages: *predicate identification*, *argument identification*, and *argument classification*. Predicate and argument identification are treated as binary classification problems. In a simple post-processing predicate classification step, a predicted predicate is assigned the most frequent sense from the training data. Argument classification is treated as a multi-class learning problem, where the classes correspond to the argument types.

3.1 Learning and Parameter Optimization

For learning and prediction we used the freely available support vector machine (SVM) implementation LIBSVM (version 2.86) (Chang and Lin, 2001). The choice of cost and kernel parameter values will often significantly influence the performance of the SVM classifier. We therefore implemented a parameter optimizer based on the DIRECT optimization algorithm (Gablonsky, 2001). It iteratively divides the search space into smaller hyperrectangles, sampling the objective function in the centroid of each hyperrectangle, and selecting those hyperrectangles that are potentially optimal for further processing. The search space consisted of the SVM parameters to optimize and the objective function was the cross-validation accuracy reported by LIBSVM.

Tests performed during training for predicate identification showed that the use of runtime optimization of the SVM parameters for nonlinear kernels yielded a higher average F_1-score effectiveness. Surprisingly, the best nonlinear kernels were always outperformed by the linear kernel with default settings, which indicates that the data is approximately linearly separable.

3.2 Filtering and Data Set Splitting

To decrease the number of instances during training, all predicate and argument candidates with PoS-tags that occur very infrequently in the training set were filtered out. Some PoS-tags were filtered out for all three stages, e.g. non-alphanumerics, HYPH, SYM, and LS. This approach was effective, e.g. removing more than half of the total number of instances for predicate prediction.

To speed up the SVM training and allow for parallelization, each data set was split into several bins. However, there is a trade-off between speed and accuracy. Performance consistently deteriorated when splitting into smaller bins. The final system contained two variants, one with more bins based on a combination of PoS-tags and lemma frequency information, and one with fewer bins based only on PoS-tag information. The three learning tasks used different splits. In general, the argument identification step was the most difficult and therefore required a larger number of bins.

3.3 Features

We implemented a large number of features (over 50)[1] for the SRL system. Many of them can be found in the literature, starting from Gildea and Jurafsky (2002) and onward. All features, except bag-of-words, take nominal values, which are binarized for the vectors used as input to the SVM classifier. Low-frequency feature values (except for *Voice*, *Initial Letter*, *Number of Words*, *Relative Position*, and the *Distance* features), below a threshold of 20 occurrences, were given a default value.

We distinguish between single node and node pair features. The following single node features were used for all three learning tasks and for both the predicate and argument node:[2]

- Lemma, PoS, and Dependency relation (DepRel) for the node itself, the parent, and the left and right sibling
- Initial Letter (upper-case/lower-case), Number of Words, and Voice (based on simple heuristics, only for the predicate node during argument classification)
- PoS Sequence and PoS bag-of-words (BoW) for the node itself with children and for the parent with children
- Lemma and PoS for the first and last child of the node
- Sequence and BoW of Lemma and PoS for content words
- Sequence and BoW of PoS for the immediate children's content words
- Sequence and BoW of PoS for the parent's content words and for the parent's immediate children
- Sequence and BoW of DepRels for the node itself, for the immediate children, and for the parent's immediate children

All extractors of node pair features, where the pair consists of the predicate and the argument node, can be used both for argument identification and argument classification. We used the following node pair features:

- Relative Position (the argument is before/after the predicate), Distance in Words, Middle Distance in DepRels
- PoS Full Path, PoS Middle Path, PoS Short Path

The *full path* feature contains the PoS-tag of the argument node, all dependency relations between the argument node and the predicate node and finally the PoS-tag of the predicate node. The *middle path* goes to the lowest common ancestor for argument and predicate (this is also the distance calculated by *Middle Distance in DepRels*) and the *short path* only contains the dependency relation of the argument and predicate nodes.

3.4 Joint Syntactic and Semantic Parsing

When considering one predicate at a time, SRL becomes a regular labelling problem. Given a predicted predicate, joint learning of syntactic and semantic dependencies can be carried out by simultaneously assigning an argument label and a dependency relation. This is possible because we know a priori where to attach the argument, since there is only one predicate candidate[3]. The MaltParser system for English described in Hall et al. (2007) was used as a baseline, and then optimized for this new task, focusing on feature selection.

A large feature model was constructed, and backward selection was carried out until no further gain could be observed. The feature model of MaltParser consists of a number of feature types, each describing a starting point, a path through the structure so far, and a column of the node arrived at. The number of feature types was reduced from 37 to 35 based on the labelled F_1-score.

As parsing is done at the same time as argument labelling, different syntactic structures risk being assigned to the same sentence, depending on which predicate is currently processed. This means that several, possibly different, parses have to be combined into one. In this experiment, the head and the dependency label were concatenated, and the most frequent one was used. In case of a tie, the first one to appear was used. The likelihood of the chosen labelling was also used as a confidence measure for the syntactic blender.

3.5 Blending and Post-Processing

Combining the output from several different systems has been shown to be beneficial (Koomen et al., 2005). For the final submission, we combined the output of two variants of the pipelined SRL system, each using different data splits, with

[1]Some features were discarded for the final system based on Information Gain, calculated using Weka (Witten and Frank, 2005).

[2]For all features using lemma or PoS the (predicted) split value is used.

[3]The version of the joint system used in the submission was based on an early predicate prediction. More accurate predicates would give a major improvement for the results.

Test set	Pred PoS	Labelled F_1	Unlabelled F_1
WSJ	All	82.90	90.90
	NN*	81.12	86.39
	VB*	85.52	96.49
Brown	All	67.48	85.49
	NN*	58.34	75.35
	VB*	73.24	91.97

Table 2: Semantic predicate results on the test sets.

	WSJ + Brown	WSJ	Brown
Syn + Sem	79.79	80.92	70.49
Syn	86.63	87.36	80.77
Sem	72.94	74.47	60.18

Table 3: Syntactic and semantic scores on the test sets for the submitted system. The scores, from top to bottom, are labelled macro F_1, labelled attachment score and labelled F_1.

the SRL output of the joint system. A simple uniform weight majority vote heuristic was used, with no combinatorial constraints on the selected arguments. For each sentence, all predicates that were identified by a majority of the systems were selected. Then, for each selected predicate, its arguments were picked by majority vote (ignoring the systems not voting for the predicate). The best single SRL system achieved a labelled F_1-score of 71.34 on the WSJ test set and 57.73 on the Brown test set, compared to 74.47 and 60.18 for the blended system.

As a final step, we filtered out all verbal and nominal predicates not in PropBank or NomBank, respectively, based on the predicted PoS-tag and lemma. Each lexicon was expanded with lemmas from the training set, due to predicted lemma errors in the training data. This turned out to be a successful strategy for the individual systems, but slightly detrimental for the blended system.

3.6 Results

Semantic predicate results for WSJ and Brown can be found in Table 2. Table 4 shows the results for identification and classification of arguments.

4 Analysis and Conclusions

In general, the mixed and blended system performs well on all tasks, rendering a sixth place in the CoNLL 2008 shared task. The overall scores for the submitted system can be seen in Table 3.

4.1 Parsing

For the blended parsing system, the labelled attachment score drops from 87.36 for the WSJ test set to 80.77 for the Brown test set, while the unlabelled attachment score only drops from 89.88 to 86.28. This shows that the system is robust with regards to the overall syntactic structure, even if picking the correct label is more difficult for the out-of-domain text.

The parser has difficulties finding the right head for punctuation and symbols. Apart from errors regarding punctuation, most errors occur for IN and TO. A majority of these problems are related to assigning the correct dependency. This is not surprising, since these are categories that focus on form rather than function.

There is no significant difference in score for left and right dependencies, presumably because of the bi-directional parsing. However, the system overpredicts dependencies to the root. This is mainly due to the way MaltParser handles tokens not being attached anywhere during parsing. These tokens are by default assigned to the root.

4.2 SRL

Similarly to the parsing results, the blended SRL system is less robust with respect to labelled F_1-score, dropping from 74.47 on the WSJ test set to 60.18 on the Brown test set. The corresponding drop in unlabelled F_1-score is from 82.90 to 75.49.

The simple method of picking the most common sense from the training data works quite well, but the difference in domain makes it more difficult to find the correct sense for the Brown corpus. In the future, a predicate classification module is needed. For the WSJ corpus, assigning the most common predicate sense works better with nominal than with verbal predicates, while verbal predicates are handled better for the Brown corpus.

In general, verbal predicate-argument structures are handled better than nominal ones, for both test sets. This is not surprising, since nominal predicate-argument structures tend to vary more in their composition.

Since we do not use global constraints for the argument labelling (looking at the whole argument structure for each predicate), the system can output the same argument label for a predicate several times. For the WSJ test set, for instance, the ratio of repeated argument labels is 5.4% in the system output, compared to 0.3% in the gold standard. However, since there are no confidence scores for predictions it is difficult to handle this in the current system.

PPOSS(pred) + ARG	WSJ F_1	Brown F_1
NN* + A0	61.42	38.99
NN* + A1	67.07	53.10
NN* + A2	57.02	26.19
NN* + A3	63.08	(16.67)
NN* + AM-ADV	4.65	(-)
NN* + AM-EXT	44.78	(40.00)
NN* + AM-LOC	49.45	(-)
NN* + AM-MNR	53.51	21.82
NN* + AM-NEG	79.37	(46.15)
NN* + AM-TMP	67.23	(25.00)
VB* + A0	81.72	73.58
VB* + A1	81.77	67.99
VB* + A2	60.91	50.67
VB* + A3	61.49	(14.28)
VB* + A4	77.84	(40.00)
VB* + AM-ADV	47.49	30.33
VB* + AM-CAU	55.12	(35.29)
VB* + AM-DIR	41.86	37.14
VB* + AM-DIS	71.91	37.04
VB* + AM-EXT	60.38	(-)
VB* + AM-LOC	55.69	37.50
VB* + AM-MNR	49.54	36.25
VB* + AM-MOD	94.85	82.42
VB* + AM-NEG	93.45	77.08
VB* + AM-PNC	50.00	(62.50)
VB* + AM-TMP	69.59	49.07
VB* + C-A1	70.76	55.32
VB* + R-A0	83.68	70.83
VB* + R-A1	68.87	51.43
VB* + R-AM-LOC	38.46	(25.00)
VB* + R-AM-TMP	56.82	(58.82)

Table 4: Semantic argument results on the two test sets, showing arguments with more than 20 instances in the gold test set (fewer instances for Brown are given in parentheses).

Acknowledgements

This project was carried out within the course Machine Learning 2, organized by GSLT (Swedish National Graduate School of Language Technology), with additional support from NGSLT (Nordic Graduate School of Language Technology). We thank our supervisors Joakim Nivre, Björn Gambäck and Pierre Nugues for advice and support. Computations were performed on the BalticGrid and UPPMAX (projects p2005008 and p2005028) resources. We thank Tore Sundqvist at UPPMAX for technical assistance.

References

Chang, Chih-Chung and Chih-Jen Lin, 2001. *LIBSVM: A library for support vector machines.*

Chu, Y. J. and T. H. Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*, 14:1396–1400.

Covington, Michael A. 2001. A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual Association for Computing Machinery Southeast Conference*, Athens, Georgia.

Edmonds, Jack. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards*, 71(B):233–240.

Gablonsky, Jörg M. 2001. *Modifications of the DIRECT algorithm.* Ph.D. thesis, North Carolina State University, Raleigh, North Carolina.

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

Hall, Johan, Jens Nilsson, Joakim Nivre, Gülşen Eryiğit, Beáta Megyesi, Mattias Nilsson, and Markus Saers. 2007. Single malt or blended? A study in multilingual parser optimization. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, Prague, Czech Republic.

Koomen, Peter, Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2005. Generalized inference with multiple semantic role labeling systems. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, Ann Arbor, Michigan.

Nilsson, Jens, Joakim Nivre, and Johan Hall. 2007. Generalizing tree transformations for inductive dependency parsing. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic.

Nivre, Joakim, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 2(13):95–135.

Nivre, Joakim. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the ACL'04 Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, Barcelona, Spain.

Surdeanu, Mihai, 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)*, Manchester, Great Britain.

Witten, Ian H. and Eibe Frank. 2005. *Data mining: Practical machine learning tools and techniques.* Morgan Kaufmann, Amsterdam, 2nd edition.