An Ensemble Model for the EVALITA 2011 Dependency Parsing Task*

Alberto Lavelli

FBK-irst,
via Sommarive 18, I-38123 Povo (TN), Italy
lavelli@fbk.eu

Abstract. This paper describes our participation in the EVALITA 2011 Dependency Parsing Task with an ensemble model. In the 2009 edition we participated with MaltParser, reusing feature models developed in the CoNLL shared task 2007. In 2011, we compared the results obtained by different parsing algorithms implemented in MaltParser with an ensemble model made available by Mihai Surdeanu. The best results were achieved by the ensemble model which was selected for the official submission. In the final evaluation, our system finished third in the dependency parsing task.
Keywords: Dependency parsing, ensemble model, Italian.

1 Introduction

In the Dependency Parsing Task of EVALITA 2011, dependency parsers are applied to an Italian dependency treebank, the Turin University Treebank (TUT[1]). Previous versions of TUT were used in 2007 and 2009 for the first two editions of the EVALITA Dependency Parsing Task [3, 2].

In 2009 we participated in the Dependency Parsing Task with MaltParser,[2] a system for data-driven dependency parsing that can be used to induce a parsing model from treebank data and to parse new data using the induced model [7]. MaltParser was one of the top performing systems in the multilingual track of the CoNLL shared tasks on dependency parsing in 2006 and 2007 [13, 6]. In the Dependency Parsing Task of EVALITA 2009, dependency parsers were applied to two different Italian dependency treebanks, the Turin University Treebank (TUT[3]) and the Italian Syntactic-Semantic Treebank (ISST-CoNLL [8]).

In the following we briefly summarize the results on TUT in 2009. Table 1 shows the results of different parsing algorithms implemented in MaltParser using 10-fold cross validation on the TUT training set. Table 2 reports the results of different parsing algorithms implemented in MaltParser on the TUT test set. The results of the two best performing systems at EVALITA 2009 are reported too. Note that the best performing

* We thank Joakim Nivre and Mihai Surdeanu for making their parsers available and for kindly answering our questions about their usage.

[1] http://www.di.unito.it/~tutreeb/

[2] Freely available at http://www.maltparser.org/

[3] http://www.di.unito.it/~tutreeb/

Table 1. Dependency parsing 2009 – main subtask: TUT; results of 10-fold cross validation.

	LAS	UAS	LA
Nivre standard	78.17	85.70	85.34
Nivre standard pseudo-projective	78.19	85.68	85.43
Nivre eager	80.65	88.42	86.97
Nivre eager pseudo-projective	80.58	88.34	86.95
Covington projective	83.27	87.70	88.96
Covington projective pseudo-projective	83.28	87.72	89.42
Covington non-projective	83.43	87.91	89.57

Table 2. Dependency parsing 2009 – main subtask: results obtained with different transition systems on the test set.

	LAS	UAS	LA
Nivre standard	81.75	90.13	86.36
Nivre standard pseudo-projective	81.56	89.73	86.36
Nivre eager	83.03	91.28	87.21
Nivre eager pseudo-projective	82.88	91.15	87.20
Covington projective	86.51	90.98	90.24
Covington projective pseudo-projective	86.48	90.98	90.77
Covington non-projective	86.50	90.88	90.96
UniTo_Lesmo @ EVALITA-2009	88.73	92.28	
UniPi_Attardi @ EVALITA-2009	88.67	92.72	

parser (UniTO Lesmo) was a rule-based parser developed in parallel with TUT and tuned on the data set. The second parser (UniPI Attardi) is a multilingual deterministic shift-reduce dependency parser that handles nonprojective dependencies incrementally and learns by means of a second-order multiclass averaged perceptron classifier. In [2] the official results for the dependency parsing task can be found. Our system obtained the third best result in the main subtask (LAS: 86.50 vs. 88.73 and 88.68 of the two best performing systems). Note that the difference between the results of the two top-performing systems was not statistically significant.

2 Participation at EVALITA 2011

As said, in 2011 we compared two approaches which will be described in the two following subsections.

2.1 MaltParser

MaltParser [12] implements the transition-based approach to dependency parsing, which has two essential components:

– A nondeterministic transition system for mapping sentences to dependency trees

– A classifier that predicts the next transition for every possible system configuration

Given these two components, dependency parsing can be performed as greedy deterministic search through the transition system, guided by the classifier. With this technique, it is possible to perform parsing in linear time for projective dependency trees and quadratic time for arbitrary (non-projective) trees [11].

Transition Systems MaltParser has four built-in transition systems:

– Nivre's arc-eager system [9]
– Nivre's arc-standard system [10]
– Covington's projective system [5]
– Covington's non-projective system [5]

The two versions of Nivre's transition system are inspired by shift-reduce parsing and use a single stack to store partially processed words. The only difference is that the arc-standard version builds trees strictly bottom-up, while the arc-eager version builds structure incrementally from left to right. In both cases, the system is restricted to projective dependency trees. Covington's system uses two stacks and can therefore process arbitrary non-projective trees, but projectivity can be enforced by restricting the use of the second stack. A more detailed description of all four systems, including proofs of correctness and complexity can be found in [11].

Classifiers Classifiers can be induced from treebank data using a wide variety of different machine learning methods, but all experiments reported below use support vector machines with a polynomial kernel, as implemented in the LIBSVM package [4] included in MaltParser. The task of the classifier is to map a high-dimensional feature vector representation of a parser configuration to the optimal transition out of that configuration. Features typically represent word forms, parts of speech and other linguistic attributes of words that appear near the top of the stack(s) or in the input buffer. For the experiments reported below, we have reused the feature representations that gave the best performance for the various transition systems in the 2007 CoNLL shared tasks.[4]

Pseudo-Projective Parsing As noted earlier, three of the four transition systems used in the experiments can in principle only derive projective dependency trees. In order to overcome this limitation, we have also experimented with pseudo-projective parsing [14], which is a technique for recovering non-projective dependencies with a strictly projective parser. First, all dependency trees in the training set are projectivized by moving arcs in the tree and encoding information about these movements using complex arc labels. A parser trained on these transformed training data will ideally learn to produce trees that are strictly projective but where some arcs have complex labels indicating that they have undergone movement. These arcs can then be moved back in a post-processing step, guided by the extended arc labels, which results in non-projective trees. This technique has been combined with the two versions of Nivre's transition system and with the projective version of Covington's system.

[4] More information about the features can be found at `http://maltparser.org/conll/conll07/`.

Table 3. Dependency parsing; results of 10-fold cross validation.

	LAS	UAS	LA
Nivre standard	81.96	86.27	88.36
Nivre standard pseudo-projective	82.24	86.59	88.90
Nivre eager	84.54	88.86	90.40
Nivre eager pseudo-projective	84.47	88.77	90.55
Covington projective	83.88	88.29	89.07
Covington projective pseudo-projective	83.97	88.36	89.85
Covington non-projective	84.16	88.56	89.97
ensemble model	85.92	90.25	91.37

2.2 Ensemble Model

The ensemble model made available by Mihai Surdeanu [16][5] implements a linear interpolation of several linear-time parsing models (all based on MaltParser). In particular, it combines five different variants of MaltParser (Nivre's arc-standard left-to-right, Nivre's arc-eager left-to-right, Covington's non projective left-to-right, Nivre's arc-standard right-to-left, Covington's non projective right-to-left) as base parsers. Each individual parser runs in its own thread, which means that, if a sufficient number of cores are available, the overall runtime is essentially similar to a single MaltParser. The resulting parser has state-of-the-art performance yet it remains very fast. We have used the ensemble more or less as it is, simply exploiting the extended models for the base parsers, which are slower but more accurate.

3 Experimental results

As in 2009, we compared the four transition systems (Nivre's arc-standard, Nivre's arc-eager, Covington's projective, Covington's non-projective), in the first three cases with and without pseudo-projective parsing, using the feature representations that produced the best results on Italian for the MaltParser system at CoNLL-2007. In addition, this year we assessed the performance of the ensemble model made available by Mihai Surdeanu.

First of all, in Table 3 we report the results obtained on the training set, used to choose the system for parsing the test set. We adopted a 10-fold cross validation on the entire training set.

Given the results shown in Table 3, the ensemble model was chosen for performing the official run. In [1] the official results for the dependency parsing task can be found. Our system obtained the third best result (LAS: 88.62 vs. 91.23 and 89.88 of the two best performing systems).

The results on the test set with the different transition systems (see Table 4) are coherent with those obtained on the training set. The comparison between the performance of the different transition systems is different from the one in 2009 when Covington algorithms performed better than Nivre eager ones. This is due to the fact that in

[5] `http://www.surdeanu.name/mihai/ensemble/`

Table 4. Dependency parsing: results obtained with different systems on the test set.

	LAS	UAS	LA
ensemble model	88.62	92.85	92.50
Nivre standard	84.83	89.30	89.69
Nivre standard pseudo-projective	85.26	89.62	90.16
Nivre eager	87.30	91.47	91.45
Nivre eager pseudo-projective	87.11	91.24	91.64
Covington projective	85.99	90.23	89.81
Covington projective pseudo-projective	86.25	90.42	90.70
Covington non-projective	86.72	90.91	90.89

2009 we made a mistake in the parameters of the Nivre algorithms that penalized their performance.

4 Conclusions

Our system cannot compete with the best performing systems on the test set. However, we consider our results fair given that we have used an off-the-shelf system and simply trained it on the Italian treebank.

To make the parsers used in our participation to EVALITA usable within other more complex NLP systems (e.g., textual entailment, question answering, ...) we are currently integrating them in the TextPro tool suite [15].

References

1. Bosco, C., Mazzei, A.: The Evalita 2011 Parsing Task: the dependency track. In: Working Notes of EVALITA 2011 (2012)
2. Bosco, C., Montemagni, S., Mazzei, A., Lombardo, V., dell'Orletta, F., Lenci, A.: Evalita'09 Parsing Task: comparing dependency parsers and treebanks. In: Proceedings of the EVALITA 2009 Workshop on Evaluation of NLP Tools for Italian (2009)
3. Bosco, C., Mazzei, A., Lombardo, V., Attardi, G., Corazza, A., Lavelli, A., Lesmo, L., Satta, G., Simi, M.: Comparing Italian parsers on a common treebank: the EVALITA experience. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation. Marrakech, Morocco (May 2008), http://www.lrec-conf.org/proceedings/lrec2008/pdf/528_paper.pdf
4. Chang, C.C., Lin, C.J.: LIBSVM: A Library for Support Vector Machines (2001), software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm
5. Covington, M.A.: A fundamental algorithm for dependency parsing. In: Proceedings of the 39th Annual ACM Southeast Conference. pp. 95–102 (2001)
6. Hall, J., Nilsson, J., Nivre, J., Eryigit, G., Megyesi, B., Nilsson, M., Saers, M.: Single malt or blended? a study in multilingual parser optimization. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007. pp. 933–939. Prague, Czech Republic (June 2007), http://www.aclweb.org/anthology/D/D07/D07-1097

7. Lavelli, A., Hall, J., Nilsson, J., Nivre, J.: MaltParser at the EVALITA 2009 dependency parsing task. In: Proceedings of the EVALITA 2009 Workshop on Evaluation of NLP Tools for Italian (2009)
8. Montemagni, S., Barsotti, F., Battista, M., Calzolari, N., Corazzari, O., Lenci, A., Zampolli, A., Fanciulli, F., Massetani, M., Raffaelli, R., Basili, R., Pazienza, M.T., Saracino, D., Zanzotto, F., Mana, N., Pianesi, F., Delmonte, R.: Building the Italian Syntactic-Semantic Treebank. In: Abeillé, A. (ed.) Building and Using syntactically annotated corpora, pp. 189–210. Kluwer, Dordrecht (2003)
9. Nivre, J.: An efficient algorithm for projective dependency parsing. In: Proceedings of the 8th International Workshop on Parsing Technologies (IWPT). pp. 149–160 (2003)
10. Nivre, J.: Incrementality in deterministic dependency parsing. In: Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together (ACL). pp. 50–57 (2004)
11. Nivre, J.: Algorithms for deterministic incremental dependency parsing. Computational Linguistics 34, 513–553 (2008)
12. Nivre, J., Hall, J., Nilsson, J.: MaltParser: A data-driven parser-generator for dependency parsing. In: Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC). pp. 2216–2219 (2006)
13. Nivre, J., Hall, J., Nilsson, J., Eryiğit, G., Marinov, S.: Labeled pseudo-projective dependency parsing with support vector machines. In: Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL). pp. 221–225 (2006)
14. Nivre, J., Nilsson, J.: Pseudo-projective dependency parsing. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL). pp. 99–106 (2005)
15. Pianta, E., Girardi, C., Zanoli, R.: The TextPro tool suite. In: Proceedings of the Sixth International Conference on Language Resources and Evaluation. Marrakech, Morocco (May 2008), http://www.lrec-conf.org/proceedings/lrec2008/pdf/645_paper.pdf
16. Surdeanu, M., Manning, C.D.: Ensemble models for dependency parsing: Cheap and good? In: Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-2010). pp. 649–652. Los Angeles, California (June 2010), http://www.aclweb.org/anthology/N10-1091