Generative re-ranking model for dependency parsing of Italian sentences

Federico Sangati

Institute for Logic, Language and Computation - University of Amsterdam
f.sangati@uva.nl

Abstract. We present a general framework for dependency parsing of Italian sentences based on a combination of discriminative and generative models. We use a state-of-the-art discriminative model to obtain a k-best list of candidate structures for the test sentences, and use the generative model to compute the probability of each candidate, and select the most probable one. We present the details of the specific generative model we have employed for the EVALITA'09 task. Results show that by using the generative model we gain around 1% in labeled accuracy (around 7% error reduction) over the discriminative model.

Keywords: dependency parsing, re-ranking, generative model.

1 Introduction

In this work we have adapted the parsing formalism described in [1], to the EVALITA'09 main dependency task. The framework was developed in order to efficiently implement and compare different probabilistic generative models of dependency structures. Probabilistic models define probabilities over all valid dependency structures of a given sentence, and they are therefore very useful for syntax based language modeling. Although recent trends in dependency parsing have shown an overall success of discriminative models (cf. [2], [3]), the current work shows that fine tuned generative models can still be competitive.

1.1 Re-ranking methodology

Our parsing framework is based on a combination of discriminative and generative models. We use the state-of-the-art Maximum Spanning Tree (MST) discriminative model in [4] to obtain a k-best list of candidate structures for the test sentences, and employ a generative model to compute the probability of each candidate, and select the most probable one. The idea of combining these two types of models is not new (cf. [5]) although earlier investigations used the generative model in the first step, and trained the discriminative model over its k-best candidates. It's important to stress that in our framework the first phase is used to generate a satisfactory and compact list of candidates in order to avoid to compute the full parsing forest for each test sentence, but the candidates list is not used as training material for the generative model.

In the training phase the generative model decomposes every annotated dependency structures into a series of independent events, each mapped to a corresponding conditioning context, and keeps track of their frequencies by building an appropriate table of events. In the re-ranking phase for every test sentence S each candidate structure T_S provided by the discriminative model is decomposed into independent events $(e_1, e_2, \ldots, e_n)$ and corresponding conditioning contexts $(c_1, c_2, \ldots, c_n)$. The probability of the structure can then be calculated as:

$$P(T_S) = \prod_{i=1}^{n} \frac{f(e_i)}{f(c_i)} \tag{1}$$

2 Implementation

2.1 The generative model

The general re-ranking framework just described allowed us to explore different generative models for dependency parsing of Italian sentences, taking as inspiration the various generative models proposed by Eisner in [6] and [7].

The TUT[1] dependency corpus of the EVALITA'09 main dependency task was composed of 2401 training sentences, and 240 raw test sentences. We have divided the annotated corpus in several 90-10 splits of training and developing sections, and run the evaluations while varying the generative model and the feature space. Figure 2.1 schematizes the generative model we finally decided to employ for the final blind test set: nodes are generated recursively in a top-down manner starting from the root. At any given node, left and right dependents are generated as two separate Markov sequences of nodes, each conditioned on the previously chosen dependent and on ancestral nodes (parent and grand parent). As in common practice we add special stop symbols after the last dependent in either direction, in order to make the probabilistic model proper and consistent. Differently from previous models (cf. [9], [7]) the Markov sequence of dependents is generated strictly left to right instead of inside-outwards.

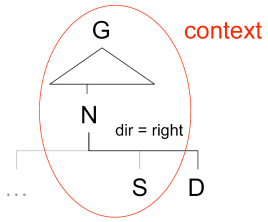

Fig. 1. A scheme of the event space and conditioning context considered in the implemented generative model. The event here is "D is a dependent of N" and its conditioning context includes the elements within the red oval.

[1] Turin University Treebank: `http://www.di.unito.it/~tutreeb` , see also [8].

2.2 Probabilistic space and features details

The probability of attaching a dependent D to a node N is generically described in equation 2. In equation 3 the context is specified in order to include information related to the parent node (N), the previously chosen sister (S) the grand parent (G) and the direction of the next dependent (dir). In equation 4 the features identifying the dependent node D are specified: *dist* represents the distance[2] of the dependent relative to the parent node, *term* is a boolean function indicating whether D is a terminal node with no more dependents, *word* is the lexical representation of node D and *tag* its Part-of-Speech (PoS). This equation is split in the 4 equations 6, 7, 8, 9, as in common practice. Each of these four equations are defined in a backoff reduction list reported in descending priority[3]. The specification of each node at any backoff level can also vary with respect to the frequency of the corresponding word in the corpus. We have in fact divided the words in three categories: closed-class words[4], frequent and infrequent open-class words. Table 2.2 reports the full details of the word backoff levels.

$$P(D|context) = \tag{2}$$

$$P(D|N,S,G,dir) = \tag{3}$$

$$P(dist(N,D), term(D), word(D), tag(D)|N,S,G,dir) = \tag{4}$$

$$\tag{5}$$

$$P(tag(D)|N,S,G,dir) \tag{6}$$

reduction list:

$D_{TF}\|N_0,S_0,G_1,dir$
$D_{TF}\|N_1,S_1,G_2,dir$
$D_{TF}\|N_0,S_3,G_3,dir$ $D_{TF}\|N_3,S_0,G_3,dir$
$D_{TF}\|N_4,S_4,G_4,dir$

$$\times P(word(D)|tag(D),N,S,G,dir) \tag{7}$$

reduction list:

$D_L\|N_0,S_1,G_2,dir$
$D_L\|N_0,S_3,G_4,dir$
$D_L\|N_4,S_4,dir$

$$\times P(term(D)|word(D),tag(D),N,S,G,dir) \tag{8}$$

reduction list:

$term(D)\|D_1,N_1,S_1,G_1,dir$
$term(D)\|D_1,N_3,S_3,G_3,dir$
$term(D)\|D_1,N_4,S_4,G_4,dir$

$$\times P(dist(P,D)|term(D),word(D),tag(D),N,S,G,dir) \tag{9}$$

reduction list:

$dist(P,D)\|D_T,N_2,S_2,G_3,dir$
$dist(P,D)\|D_T,N_3,S_3,G_4,dir$
$dist(P,D)\|D_T,N_4,S_4,G_4,dir$

[2] The distance function returns 4 values according to the 4 ranges: 1, 2, 3-6, 7-∞.

[3] All the notation and backoff parameters are identical to [6].

[4] A closed-class word must have a tag within the following list: ART, CONJ, PHRAS, PREDET, PREP, PRON, PUNCT, SPECIAL.

Table 1. Words backoff levels, for closed-class and frequent/infrequent open-class.

Level	Closed-class	Open-class (freq $\geq$ 4)	Open-class (freq $<$ 4)
0	lex+PoS+features	lex+PoS+features	lemma+PoS
1	lex+PoS+features	lemma+PoS+features	lemma+PoS
2	lex+PoS+features	lemma+PoS	lemma+PoS
3	lemma+PoS+features	PoS+features	PoS+features
4	PoS	PoS	PoS
TF	PoS+features		
T	PoS		
L	lex		

3 Results

Table 2 reports the results we obtained with the MST discriminative model and by applying the generative re-ranking model over its 10-best candidates. Both labeled and unlabeled scores improved around 1% point over the discriminative model. To have a more detailed investigation over the results, we show in figure 2 the labeled attachment score of the generative model for each category of the dependents, and in figure 3 the absolute number of improvements for each category with respect to the discriminative model. The increases in accuracy of the generative model, although relatively modest, cover almost all categories including the four most difficult ones: PREP, PUNCT, PRON, and CONJ.

4 Conclusions

We have presented a general framework for dependency parsing based on a combination of discriminative and generative models. This framework allowed us to compare several probabilistic generative models, to choose the most promising one and to select its most appropriate feature space[5]. Results showed that by using the generative model we can gain around 1% in labeled accuracy over the results obtained using the discriminative model alone. One open question is how much the particular annotation style prevents the parser from recovering the correct structure of particular constructions. Previous literature (cf. [10]) has shown that different head annotation schemes[6] could lead to better results in parsing.

Acknowledgments We gratefully acknowledge funding by the Netherlands Organization for Scientific Research (NWO): the author is funded through a Vici-grant "Integrating Cognition" (277.70.006) to Rens Bod.

[5] In the re-ranking phase we have not used any information concerning the labels of the dependencies, but only the unlabeled structure of the tree. It would be therefore interesting to include this information in future settings, especially since labeled accuracy is considered the most significant evaluation metrics.

[6] For instance the conjunctions could be the head of coordinations instead of the first conjuncts, and the verbs the head of sub-clauses instead of the complementizers.

Table 2. Overall results of the MST model and our re-ranking generative models, as labeled and unlabeled attachment scores.

	UAS	LAS
MST 1-best	91.36	83.90
Reranked	**92.60**	**84.98**
Improvement	1.25	1.08
Error reduction	14.44	6.70

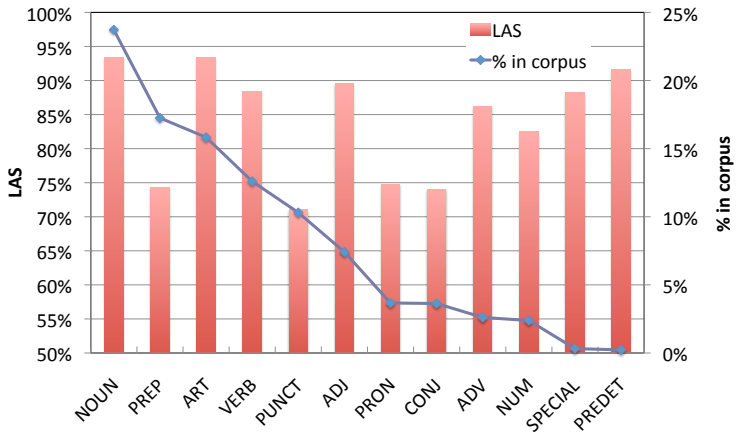

Fig. 2. Labeled attachment score of the re-ranking generative model within the different PoS-tags of the dependents.

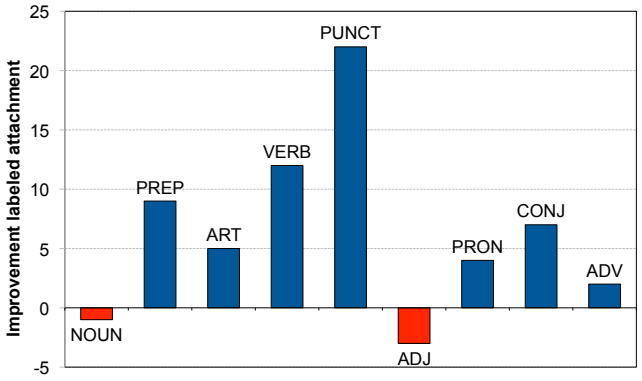

Fig. 3. Absolute number of improvements of the re-ranking generative model over the discriminative model for the 9 most frequent PoS-tags in the corpus.

References

1. Sangati, F., Zuidema, W., Bod, R.: A generative re-ranking model for dependency parsing. In: Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09), pp. 238–241. Association for Computational Linguistics (2009)
2. Buchholz, S., Marsi, E.: CoNLL-X shared task on multilingual dependency parsing. In: Proceedings of CoNLL, pp. 149–164 (2006)
3. Nivre, J., Hall, J., Kübler, S., Mcdonald, R., Nilsson, J., Riedel, S., Yuret, D.: The CoNLL 2007 shared task on dependency parsing. In: Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007, pp. 915–932.Association for Computational Linguistics (2007)
4. McDonald, R.: Discriminative learning and spanning tree algorithms for dependency parsing. PhD thesis. Philadelphia, PA, USA (2006)
5. Collins, M., Duffy, N.: New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In: Proceedings of 40th Annual Meeting of the Association for Computational Linguistics, pp. 263–270. Association for Computational Linguistics (2002)
6. Eisner, J.M.: An empirical comparison of probability models for dependency grammar. Technical Report IRCS-96-11, University of Pennsylvania (1996)
7. Eisner, J.M.: Three new probabilistic models for dependency parsing: an exploration. In: Proceedings of the 16th conference on Computational linguistics, pp. 340–345. Association for Computational Linguistics (1996)
8. Lesmo, L., Lombardo, V., Bosco, C.: Treebank development: the TUT approach. In: Proceedings of the International Conference on Natural Language Processing, pp. 61–70. Vikas Publishing House (2002)
9. Collins, M.J.: Head-driven statistical models for natural language parsing. PhD thesis, University of Pennsylvania (1999)
10. Sangati, F., Zuidema, W.: Unsupervised methods for head assignments. In: Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009), pp. 701–709. Association for Computational Linguistics (2009)