DESR NEL TASK DI EVALITA PER PARSER A DIPENDENZE

DESR AT THE EVALITA DEPENDENCY PARSING TASK

GIUSEPPE ATTARDI · MARIA SIMI

SOMMARIO/ *ABSTRACT*

DeSR è un parser a dipendenze shift/reduce multilingua in grado di trattare incrementalmente le dipendenze non proiettive. Apprende da corpora annotati le azioni per costruire gli alberi di parsing. Per Evalita, DeSR usa come algoritmo di apprendimento un classificatore del second'ordine multiclasse mediato basato su perceptron.

DeSR is a multilingual deterministic shift/reduce dependency parser, capable of handling non-projective dependencies incrementally. It learns from annotated corpora the actions to use for building the parse trees. For the Evalita task DesR used a second-order multiclass averaged perceptron classifier as a learning algorithm.

Keywords: Dependency parsing, shift/reduce parsing

1. Introduction

Dependency-based syntactic parsing is the task of uncovering the dependency tree of a sentence, which consists of labeled links representing dependency relationships between sentence tokens. Dependency parsing can be cast into a series of classification tasks, choosing whether and which label to assign to a pair of tokens. Classification can be dealt by learning from annotated corpora, which are becoming more generally available for several languages ([4], [9]).

Parsing algorithms differ in the way pairs of tokens are considered. An approach is to use a grammar to generate possible trees. Among the algorithms with dispense with the use of grammars the two most typical examples are:

1. *graph-based parsers* learn a scoring function over the set of possible parses. Parsing is done by generating spanning trees as candidates and selecting the one with the highest score [6];
2. *transition-based parsers* use training data to learn a process for building a dependency graph [10].

The latter approach can lead to deterministic and incremental dependency parsers like those in [7] and [1].

Deterministic parsing of natural language is motivated both by psycholinguistic concerns and by the performance requirements of possible applications in text mining on large Web collections.

2. The DeSR Dependency Parser

DeSR (Dependency Shift Reduce) is a transition-based parser which is both deterministic, i.e. it chooses a single action to perform at each step, and incremental, i.e. it does not have to analyze the sentence repeatedly. DeSR performs both attachment and labeling in one step and deals with non-projective dependencies by means of specific parsing rules [1]. A tree revision stage, deterministic as well, can be added for correcting parser mistakes or ambiguous sentences [3].

The parser constructs dependency trees by scanning input sentences in left-to-right word order and performing *Shift*/*Reduce* parsing actions. The state of the parser is represented as a quadruple $\langle S, I, T, A\rangle$, where I is the sequence of remaining input tokens consisting initially of n pairs (w_i, p_i) of a word and associated features. S is the stack containing analyzed tokens; T is a stack of tokens whose processing has been delayed; A is a set of labeled arc dependencies constructed so far.

At each step in the algorithm [1], the parser applies a classifier to the features representing its current state in order to select a parsing rule to modify its state.

Several classifiers are available, including SVM, Maximum Entropy, Memory-Based Learning and Perceptron. For Evalita we used a multiclass regularized second-order averaged perceptron [5].

The three basic parsing rule schemas are:

$$\textit{Shift} \quad \frac{\langle S, n|I, T, A\rangle}{\langle n|S, I, T, A\rangle}$$

$$\textit{Right}_d \quad \frac{\langle s|S, n|I, T, A\rangle}{\langle S, n|I, T, A\cup\{(s, d, n)\}\rangle}$$

$$\textit{Left}_d \quad \frac{\langle s|S, n|I, T, A\rangle}{\langle S, s|I, T, A\cup\{(n, d, s)\}\rangle}$$

The schemas for the *Left* and *Right* rules are instantiated for each dependency type d.

Additional parsing rules are included for dealing with non-projective links: rules *Left2*, *Right2* are similar to *Left* and *Right*, except that they create links crossing one intermediate node, while *Left3* and *Right3* cross two intermediate nodes. Finally rules *Extract*/*Insert* generalize the previous rules by respectively moving one token to the temporary stack *T* and reinserting the top of *T* into *S*.

These rules proved more effective in handling non-projective than the approach of pseudo-projective parsing which involves deprojectivization by means of pre/post-processing steps [8].

In the TUT treebank, the rules occur with these frequencies in the train set: Left2 80, Right2 14, Left3 16, Right3 0. They never occur though in the test set.

2.1 Feature selection

In DeSR, the set of features to use in parsing is configurable through a parameter file and consists in a choice of tags from various tokens, in particular form, lemma, POS, morphology and dependency label.

The set of features used in the evaluation was chosen trough a process of feature selection, starting from a fairly comprehensive set of 40 features and trying all variants obtained by dropping a single feature, as described in [1].

3. Experimental results

In Table 1 we report the Unlabeled Attachment Score (UAS) and the Labeled Attachement Score (LAS) obtained on the Evalita Dependency Parsing Task. We also report parsing times on the test sets, expressed in seconds on a 2.4 GHz Xeon server, with 4 GB of memory.

Table 1: Accuracy on the Dependency Parsing Task

Test set	LAS	UAS	Time
codciv	79.13	91.37	26.65
newspap	76,62	85,49	22.86
DeSR average	77,88	88,43	24.75
Best average	**86.93**	**90.90**	

The large difference in accuracy between LAS and UAS indicates a significant number of labeling errors, so we tried reducing, both in the training and the test data, the number of dependency labels by abstracting 31 types out of the initial set of 74. With this reduced set of labels, which is more in line with the number of dependency types usually found in other corpora of comparable dimensions, we achieved an average of 83.27% LAS and 88.54% UAS.

4. Conclusions

The DeSR parser was trained on the TUT treebank using a similar setting as that used to handle the ISST Italian corpus provided as part of the CoNLL 2007 Shared Task. The results we obtained in the CoNLL competition on the ISST corpus are: 81.34% (LAS) and 85.54% (UAS).

The Evalita official results on the TUT corpus are better in UAS and worse in LAS. This is due to the large number of dependency types and a relatively small set of examples for each type. With a smaller set of labels we obtain better results in the LAS score as well.

Overall the DeSR parser achieved the best accuracy among the statistical parsers, showing that it can adapt quite well to different corpora.

REFERENCES

[1] G. Attardi. 2006. Experiments with a Multilanguage non-projective dependency parser. In *Proc. of the Tenth CoNLL*

[2] G. Attardi, F. Dell'Orletta, M. Simi, A. Chanev and M. Ciaramita. 2007. Multilingual Dependency Parsing and Domain Adaptation using DeSR. *Proc. of CoNLL 2007 Shared Task EMNLP-CoNLL 2007*, Prague.

[3] G. Attardi, M. Ciaramita. 2007. Tree Revision Learning for Dependency Parsing. In *Proc. of NAACL/HLTC 2007*.

[4] S. Buchholz, et al. 2006. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proc. of the Tenth CoNLL*.

[5] M. Ciaramita, G. Attardi. 2007. Dependency Parsing with Second-Order Feature Maps and Annotated Semantic Information. *Proc. of the 10th International Conference on Parsing Technologies* (*IWPT*).

[6] R. McDonald, F. Pereira, K. Ribarov, and J. Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proc. of Conference HLT-EMNL*.

[7] J. Nivre and M. Scholz. 2004. Deterministic Dependency Parsing of English Text. In *Proc. of COLING 2004*, Geneva, Switzerland, 64–70.

[8] J. Nivre and J. Nilsson. 2005. Pseudo-Projective Dependency Parsing. In *Proc. of the ACL*, 99–106.

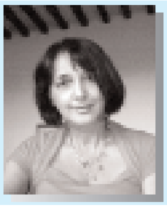
[9] J. Nivre, et al. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proc. of the CoNLL 2007 Shared Task. Joint Conf. on EMNLP CoNLL*.

[10] H. Yamada, Y. Matsumoto. 2003. Statistical Dependency Analysis with Support Vector Machines. *Proc. of the 8th IWPT*, 195–206.

CONTACTS

GIUSEPPE ATTARDI, MARIA SIMI
Dipartimento di Informatica, Università di Pisa - Email: {attardi, simi}@di.unipi.it

GIUSEPPE ATTARDI is Full Professor at the Università di Pisa (Dip. di Informatica). He is spending a sabbatical year at Yahoo! Research, Barcelona. His research interests include Text Mining and NL Technology.



MARIA SIMI is Associate Professor at the Dipartimento di Informatica of the Università di Pisa. Her main research interests include Human-Computer Interaction and Natural Language Technology.