Task-oriented Dependency Parsing Evaluation Methodology

Alexander Volokh
DFKI
Stuhlsatzenhausweg 3
66123 Saarbrücken, Germany
alexander.volokh@dfki.de

Günter Neumann
DFKI
Stuhlsatzenhausweg 3
66123 Saarbrücken, Germany
neumann@dfki.de

Abstract

Traditional parser evaluation with attachment scores is not helpful for researchers who want to find the most suitable parser for their application. First, because it is being done for a domain which is almost always different from the domain of the application and second because many of the tested dependencies are irrelevant for the application. The alternative extrinsic evaluation is problematic as well, since it is difficult to find a suitable data set and because it is not straightforward how to measure the quality of the parser in the context of a broader appllication. We propose a method which combines the strengths of attachment scores and extrinsic evaluation and avoids their weaknesses. On the one hand we use the very robust attachment scores, We apply our approach to RTE-7 data in order to demonstrate how it works.

1 Introduction

Dependency parsers have recently become extremely popular thanks to their ability to capture the structure of a sentence in a very transparent way. Furthermore, dependency parsing algorithms work for many languages, including those with flexible word order. Therefore they have been being used in numerous fields of natural language processing, such as machine translation, information extraction, question answering, summarisation or generation and many others. In the course of the last years numerous different dependency parsers have been developed. Among them are MaltParser, MST Parser, Stanford Parser, MINIPAR, Ensemble, MDParser. Certainly there are many more dependency parsers not in this list and thus the diversity of choice makes it difficult to decide which parser is the most suitable for one's needs.

For many NLP tasks, such as MT being the extreme example, their evaluation is at least as complex as the task itself. For dependency parsing for a long time it was not the case. In fact one of the most advocated advantages of dependency parsing was the facility of its evaluation. The easiness of dependency parsing evaluation consisted in that a dependency tree can be represented as a set of individual dependency relations and the percentage of correctly assigned relations could be thus computed in a very simple fashion. The proportion of correctly recognised head-modifier dependencies is called unlabeled attachment score (UAS). Since dependency relations are usually typed, the percentage of correctly recognised dependency relations including the type (e.g. subject relation) is called labeled attachment score (LAS).

Especially in the years 2007-2009, when the CoNLL-X shared tasks in dependency parsing (cf. Buchholz and Marsi, 2006) took place, the metric became predominant in the field. It allowed to compare and rank the results of different systems for the same data set. The analysis could be deepened to a comparison of performance for different dependency types, sentence lengths and further more fine-grained evaluation.

In the CoNLL shared tasks the attachment scores (AS) have proven themselves useful, since in these tasks the ability of parsers to replicate the gold standard was measured. However, for the everyday use in NLP applications the AS turned out to be less practicable. There are numerous reasons for that:

1. AS treat all dependencies equally. However, for some applications certain dependencies are more relevant than the others. In general semantically meaningful dependencies, such as subject, object, temporal or locational dependencies, are more important than some pure syntactical ones, such as punctuation or determiners.

2. The CoNLL AS have been computed for the very same data (e.g the same Penn Treebank section for English) over the years. It is unclear which parsers have rather been optimised to the data and which ones can perform well independently of the domain.
3. Especially due to the fact, that the dependency treebanks contain a certain amount of annotation errors (cf. Meurers et al.,2008; Volokh and Neumann, 2011), the latest improvements in one percent range are questionable and new methods of evaluating progress are necessary.

Therefore for those who primarily use dependency parsers in applications there is a growing interest in not treating parsing as an independent task, but rather to evaluate parser performance in a n application, ideally the same they are going to be used in. However, this so called extrinsic evaluation, is not flawless neither. Even though different parsers can be easily interchanged in the same system and their impact on the final result can be compared, this strategy has the following drawbacks:
1. It is unclear whether the impact is due to parser quality or due to the quality of the embedding, and what is the relation between both.
2. Depending on the task the embedding might be very difficult, both timewise and regarding the required level of expertise.

In this paper we propose such a methodology which allows to evaluate dependency parsers in an application and present our results of applying it to the task of recognising textual entailment (RTE). Our strategy combines benefits of intrinsic and extrinsic evaluation approaches and at the same time tries to avoid their disadvantages.

2 Related Work

The criticism of AS is not new and there is a lot of different work regarding this topic. We will not try to present a complete overview in this paper, but we will try to point out the main directions of research in this area.

First, there are numerous papers which deal with the problem of equality of dependency relations in UAS/LAS evaluation. The proposed solution to the problem is to restrict the evaluation only to difficult and/or useful relations, such as unbounded dependencies (Nivre et al., 2010) or non-local dependencies (Bender et al., 2011).

A different possibility to evaluate parsers is to compare their performance in an embedding task, e.g. by incorporating dependency relations as statistical features in the task-specific system. Depending on the quality of the parsers the accuracy improvements, after including such features, will vary. Examples of this method include parser evaluation for information extraction (Miyao et al., 2009; Buyko and Hahn, 2010) or even textual entailment (Yuret et al., 2010; Volokh and Neumann, 2010).

3 Related Work Analysis

Whereas the mentioned work basically addresses all the weaknesses of AS, there is no methodology, which copes with all of them at once. Thus whereas the idea of considering only a subset of dependency relations is great, there is still no guarantee that these relations are useful for one's own research purposes. Similarly, whereas the inclusion of dependency-based features seems to be promising, since the machine learning classifier will learn what is useful automatically, it becomes difficult to differentiate between the quality of dependency relations and the quality of the component making use of them. Let us exemplify the latter on the basis of the both already mentioned work for parser evaluation using textual entailment.

Textual entailment is a relation between text fragments, which states whether the meaning of one fragment is contained in the other one. The entailing text fragment is usually called text (T), the entailed fragment is usually called hypothesis (H), and both are usually referred to as T/H pair.

Yuret et al. proposed a method for construction of T/H pairs for subsequent judgement whether T entails H or not. These pairs are constructed in such a way that the entailment relation can be predicted properly only in case when the necessary dependency relations were classified correctly. Here are examples for such dependency relations:

subject-verb dependency: "John kissed Mary." entails "John kissed somebody."

verb-object dependency: "John kissed Mary." entails "Mary was kissed."

noun-modifier dependency: "The big red boat sank." entails "The boat was big."

These examples show that entailment can be determined only if the parser correctly determined

the subjects, the objects and the noun modifiers, respectively.

On the one hand it is indeed a very good method for testing whether one's parser is able to recognise those important dependencies correctly. However, one has to keep in mind that it is not the parser accuracy being evaluated, but the RTE system. Therefore it is important to have a good module for making use of parser results in order to predict the entailment relation correctly, and this is not a trivial task. If T entails H, some dependency relations of H have to match those occurring in T. However, sometimes the matching is not straightforward like subject(T) = “John” and subject(H) = “John”, but can be trickier like object(T) = “Mary” and object(H) = “somebody” or even subject(T)=”I” and subject(H)=”woman” (in T = "Oh," said the woman, "I've seen that picture already."; H = The woman has seen something.)[1]. Additionally, when applying to real-world data in most cases synonymy and/or semantic relatedness becomes important, such as in the following T/H pair:

T = Pet owners were forced to abandon their animals in the midst of evacuation.

H = People were forced to leave their pets behind when they evacuated New Orleans.

Here the following dependencies have to match: subject(T) = “pet owners” and subject(H) = ”people”, as well as vc(T) = “abandon” and vc(H) = “leave” and object(T) = “animals” and object(H) = “pets”.

Additionally, the problem with this method is that different T/H pairs can have completely different levels of difficulty, e.g. as far as the number of correctly recognised relevant dependencies is concerned, but they contribute equally to the score. Finally, sometimes the correct decision is made simply by chance (only two possible classes YES and NO, with good chances of guessing) despite the wrongly predicted dependency structure.

The work by Volokh and Neumann shows that for real-world textual entailment data of the RTE-6 challenge (Bentivogli et al., 2010) it is very difficult to achieve a high f-score using syntactic dependencies, not only because they might be incorrectly predicted by the parsers, but because the module for matching becomes too complex, since all sorts of knowledge, including, but not limited to lexical semantics, coreference resolution, logic and inference, world and domain knowledge become necessary.

Eventually, all of the above work does not help a researcher, who wants to find the most suitable parser for his own application, if there is no application-specific study for it. Whereas the usual alternative in this case is to fall back on the task-independent parser evaluation, such as AS, we think that they lack conclusiveness about the parser suitability for one's needs and propose a different methodology.

4 Proposed Methodology

Having analysed the strengths and weaknesses of the existing evaluation methods we decided to develop an alternative, combining their positive aspects and avoiding their disadvantages.

On the one hand we thought that it is essential for the evaluation to be task-specific, since it is a perfect possibility to find out whether a parser is suitable for the given task or not. At the same time, we believed that it is a great idea to restrict the evaluation set of dependencies only to the important ones. However, to our mind there is no universal set of important relations, because for one task one set of relations might be relevant and for another task it could be a completely different one. Additionally, we wanted to avoid the embedding into a broader NLP application context, typically done in extrinsic evaluation, since then it becomes difficult to differentiate between the quality of the dependencies and the quality of the embedding.

The resulting methodology looks like this:

1. Identify the relevant tokens (words) for the given task (cf. Yuret et al. with the necessary dependency relations for recognising textual entailment in the previous section).
2. Annotate these tokens with the desired dependency relations.
3. Parse the data.
4. Compare the output of the parsers with the manual annotation.

Our proposed methodology is thus a combination of intrinsic and extrinsic methods. On the one hand we perform a task-specific evaluation, however, instead of embedding the parsers into an application we evaluate on the level of grammatical relations. Additionally, we only perform our evaluation for the important tokens and therefore the

[1]All examples are real examples from the PETE shared task.

overall score is not distorted by the average for all tokens.

The obvious disadvantage is that the annotation has to be done manually and in theory it requires the knowledge of the dependency grammar representation. However, in practice the overwhelming majority of dependency relations between the relevant tokens is of simple nature, since they belong to such easy-to-annotate types as subjects, objects or modifiers. In any case, from our experience, since we have done both in our former work, the task of annotation requires much less expertise than the task of embedding of dependency relations into an NLP system. Furthermore, the annotation process can be semi-automated, e.g. by initialising the annotation of all identified tokens with the one proposed by some parser. The actual annotation process is then reduced to the manual correction of the latter, which is usually much less work than providing the annotation from scratch.

5 Parser Evaluation for the RTE Task

We have applied our methodology for a small part of RTE-7 (Bentivogli et al., 2011) development data. We have processed 100 positive T/H pairs (from 1136 total). For these 100 pairs we have taken the corresponding 100 hypotheses and applied our algorithm. We took positive pairs, because they always overlap in meaning, on the contrary to the negative pairs which sometimes were completely unrelated to each other. Because the negative pairs account for the overwhelming majority (>95%) it would have unnecessarily complicated the annotation process, especially because we could not even annotate all the available positive ones. We did not apply the strategy to both texts (Ts) and hypotheses (Hs), but rather only hypotheses, since both T and H of the same T/H pair usually contain very similar dependencies and it would require the double effort in order to obtain the double amount of approximately identical material. It took us only several days to create this data.

Furthermore, it is important to note that Hs could not be taken independently of the T/H pair they occur in, since the set of relevant tokens in H depends on the particular T. E.g. consider the following H = "Christine O. Gregoire has been elected Governor." This H is entailed by the following Ts: T_1 = "Christine O. Gregoire, the Democratic attorney general, last week was declared the winner."; T_2 = "But for now, Gregoire remains scheduled to take the oath of office, and she insists she will do so."; T_3 = "Fifty-eight days after the election, Christine O. Gregoire was declared the governor-elect of Washington on Thursday".

When identifying the relevant tokens in H in combination with T_1 one does need the information that it was a gubernatorial election. For T_2, in addition to that, the first name becomes irrelevant. In contrast to that, for T_3 all tokens are relevant.

Overall, the analysed 100 hypotheses consisted out of 1058 tokens. 664 (62.8%) of them were marked as relevant. Eventually, the 664 tokens were annotated with a manually created gold standard and then compared with the results produced by two state of the art dependency parsers: MST Parser (88.4 LAS) and MaltParser (85.6 LAS). The relevant dependencies were of the following types: {OPRD=10, NAME=48, LGS=7, IM=7, TMP=9, AMOD=3, OBJ=34, DIR=2, SBJ=99, ADV=25, DEP=2, LOC=25, PMOD=88, VC=36, CONJ=2, SUB=2, PRD=42, COORD=3, MNR=1, ROOT=87, APPO=9, NMOD=123}[2].

The evaluation shows some interesting facts. First, the MST Parser and MaltParser, which achieve almost identical results for the standard CoNLL test data perform differently for the RTE-7 data or at least for the fraction that we have selected. Thus the method helped to find a more suitable parser for the task, where the traditional evaluation would not suffice. Second, the analysis demonstrates that only half of the data is relevant and requires a correct dependency analysis. It does not matter how the parser performs for the rest of the data. The relevant relations belong to a very small subset of all relations present in the data (overall there are more than 40 different types). The most important ones are the main predicates of the sentence (ROOT, VC or PRD), as well as subjects, objects, locations and modifiers.

We have also performed several experiments on the relevant dependencies in the CoNLL test data:

First, we have examined how many dependencies belong to the set of relevant ones. Only ~73% of all dependencies are of one of the relevant types, which again confirmed our argument that an average over all tokens is not appropriate.

[2]The label names are explained in Surdeanu et al., 2008

Second, we have evaluated the performance of MaltParser and MSTParser only for the relevant types and found out that a) MaltParser performed better (92.5% LAS MaltParser, 91.7% LAS MST Parser) for the CoNLL data and b) the performance is higher than the average over all dependencies (it is around 90% for both parsers). Both points also support our thesis that an evaluation on a different domain is not transferable to the desired application domain: a) demonstrates that the traditional CoNLL evaluation on the standard test data would not help selecting the most appropriate parser for the task. The point b) additionally demonstrates how the performance of a parser drops as soon as the domain of the application is not the same as the one the parser was trained on and that despite the fact that the relevant dependencies even seem to be easier than average, because of the higher scores for them compared to the overall score over all dependency types.

6 Discussion

The most problematic part about this approach is the determination of what is a relevant or a necessary token. It is quite easy in case of the PETE shared task data, where each T/H pair aims at evaluation of only one necessary dependency relation per T/H pair and the same words in both T and H are used. However, for real-world Ts and Hs, selected out of newspaper texts, it is much more difficult, because in most cases numerous dependencies expressed with different words are necessary.

Therefore in our work we rather speak of relevance (cf. Anderson et al., 1992). The general idea is that given some logic formulae and a task, relevant parts are those shared by the formulae, e.g. atomic units like variables and constants, whereas the necessary parts are those actually required to solve the task. E.g. for argument validation, the information contained in the premises should also occur in the conclusion in order to be relevant, but only some of such premises provide (or take away) support and are necessary in order to decide the validity. The same idea can be transferred to the information-based textual tasks: the relevant fragments are those which share the same information, whereas the necessary ones are those which are actually required for task purposes, as determining the entailment relation.

The exact distinction between relevant and necessary dependencies in a certain application is difficult and depends on how they are used in order to solve the actual task. However, it is obvious that at least as far as the evaluation of dependency parsing is concerned, it is already very sensible to discard the irrelevant dependencies. Even though a further distinction between necessary dependencies and relevant ones would be a plus, the first step is already a big improvement.

7 Conclusion

Traditional dependency parser evaluation with attachment scores is often not very helpful for researches who want to find the most suitable parser for their application. The main reasons are that the dependencies being tested are often irrelevant for the task and the domain of the test data differs from the domain of the application. The alternative extrinsic evaluation is problematic as well. On the one hand, it is difficult to find a suitable data set for one's application, and on the other hand it is not straightforward how to measure the quality of a parser in the context of a broader application.

We proposed a method which allows a satisfactory parser evaluation for the domain of the application and only for the relevant dependencies, such that the score is not distorted by the average over all tokens. Eventually, we apply our method to the RTE-7 data and present our findings. On the one hand one can see the difference between parsers, which is invisible in the traditional evaluation. On the other hand one can clearly see what kind of dependencies are relevant for the task and their numbers, which clearly demonstrates that considering all tokens would have distorted the result greatly.

8 Acknowledgements

The work presented here was partially supported by a research grant from the German Federal Ministry of Education and Research (BMBF) to the DFKI project Deependance (FKZ. 01IW11003).

References

Anderson, Alan Ross, Belnap, Nuel D. and Dunn, J. Michael, *Entailment. The logic of relevance and necessity. Vol. II* (Princeton, NJ: Princeton University Press, 1992). With contributions by Kit Fine, Alas-

dair Urquhart et al, Includes a bibliography of entailment by Robert G. Wolf

Bender, Emily M., Dan Flickinger, Stephan Oepen and Yi Zhang. 2011. *Parser Evaluation over Local and Non-Local Deep Dependencies in a Large Corpus.* Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, Edinburgh,UK.

Bentivogli L.,Clark P., Dagan I., Dang H. T. and Giampiccolo D. 2010. *The sixth PASCAL recognizing textual entailment challenge.* In The Text Analysis Conference (TAC 2010).

Bentivogli L.,Clark P., Dagan I., Dang H. T. and Giampiccolo D. 2011. *The seventh PASCAL recognizing textual entailment challenge.* In The Text Analysis Conference (TAC 2011).

Sabine Buchholz and Erwin Marsi. 2006. *CoNLL-X shared task on multilingual dependency parsing*. In Proceedings of CONLL-X, pages 149–164, New York.

Ekaterina Buyko and Udo Hahn. *Evaluating the Impact of Alternative Dependency Graph Encodings on Solving Event Extraction Tasks*. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 982-992, Association for Computational Linguistics, 2010

Detmar Meurers, Adriane Boyd and Markus Dickinson, 2008. *On Detecting Errors in Dependency Treebanks*. In Research on Language and Computation, vol. 6, pp. 113-137.

Miyao, Yusuke, Kenji Sagae, Rune Sætre, Takuya Matsuzaki and Jun'ichi Tsujii. *Evaluating Contributions of Natural Language Parsers to Protein-Protein Interaction Extraction***.** Bioinformatics. 25(3). pp. 394-400, Oxford University Press, February 2009.

Joakim Nivre, Laura Rimell, Ryan McDonald and Carlos Gómez-Rodríguez. 2010. *Evaluation of Dependency Parsers on Unbounded Dependencies*. Proceedings of the 23rd International Conference on Computational Linguistics (COLING-10), Beijing, China.

Alexander Volokh and Günter Neumann, 2010. *372:Comparing the Benefit of Different Dependency Parsers for Textual Entailment Using Syntactic Constraints Only.* In Proceedings of the SemEval-2010 Evaluation Exercises on Semantic Evaluation.

Alexander Volokh and Günter Neumann. 2011. *Automatic Detection and Correction of Errors in Dependency Treebanks* , In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp.346-350.

Deniz Yuret, Aydın Han and Zehra Turgut, 2010. *SemEval-2010 Task 12: Parser Evaluation using Textual Entailments.* In Proceedings of the SemEval-2010 Evaluation Exercises on Semantic Evaluation.