Dependency Parsing

N. Green
Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics, Prague Czech Republic.

Abstract. Dependency parsing has been a prime focus of NLP research of late due to its ability to help parse languages with a free word order. Dependency parsing has been shown to improve NLP systems in certain languages and in many cases is considered the state of the art in the field. The use of dependency parsing has mostly been limited to free word order languages, however the usefulness of dependency structures may yield improvements in many of the word's 6,000+ languages.
I will give an overview of the field of dependency parsing while giving my aims for future research. Many NLP applications rely heavily on the quality of dependency parsing. For this reason, I will examine how different parsers and annotation schemes influence the overall NLP pipeline in regards to machine translation as well as the the baseline parsing accuracy.

Introduction

Dependency parsing has been shown to be an important part of many NLP applications. Contrary to it's counterpart constituency structure, dependency tree structure is considered state-of-the-art and more useful in free word order languages. A common problem parsers have in these languages is the phenomenon of non-projectivity. This is when a subtree of a dependency graph is not contiguous, or visually cannot be drawn without intersecting lines(Kuhlmann and Satta [2009]). Dependency structures are better at showing agreement whereas constituency, or phrase based, trees typically show neighboring node groupings better due to the divide and conquer approach that context free grammars impose on sentences. My research into dependency parsing will continue on a few different paths as described in the sections below.

Data

Much of the current progress in dependency parsing has been a result of the availability of common data sets in a variety of languages, made available through the CoNLL shared task. This data is in 13 languages and 7 language families. Later shared tasks also released data in other genres to allow for domain adaptation (Nivre et al. [2007a]). The availability of standard competition, gold level, data has been an important factor in dependency based research.

Metrics

As an artifact of the CoNLL shared tasks competition, two standard metrics for comparing dependency parsing systems emerged. *L*abeled *a*ttachment *s*core (LAS) and *u*nlabeled *a*ttachment *s*core (UAS). UAS studies the structure of a dependency tree and assesses whether the output has the correct head and dependency arcs. In addition to the structure score in UAS, LAS also measures the accuracy of the dependency labels on each arc. A third, but less common metric, is used to judge the percentage of sentences that are completely correct in regards to their LAS score. This score is better used to judge how a dependency parser will affect other NLP tools that make use of the dependency parser output (Buchholz and Marsi [2006]).

To evaluate machine translation results we will rely on the Bleu (*Bi*L*ingual* *E*valuation *U*nderstudy) Metric. Bleu is an automatic scoring mechanism for machine translation that is quick and can be reused as a benchmark across machine translation tasks. BLEU is based on the geometric mean of n-gram precisions comparing a machine translation and a reference text (Papineni et al. [2002]).

Dependency Parsing Techniques

In Kübler et al. [2009] the authors confirm that two parsers, MST parser and Malt parser, give similar accuracy results but with very different errors. MST parser, a maximum spanning tree graph-based algorithm, has evenly distributed errors while MaltParser, a transition based parser, has errors on mainly longer sentences. This result comes from the approaches themselves. MST parser is globally trained so the best mean solution should be found, this is why errors on the longer sentences are about the same as the shorter sentences. Malt Parser on the other hand uses a greedy algorithm with a classifier that chooses a particular transition at each vertex. This leads to the possibility of the propagation of errors further in a sentence (McDonald and Nivre [2007]). Both these algorithms are discussed below along with a third technique, constituent transformation. It is important for all future empirical experiments to look at each kind of parser as the different types of errors may greatly change the resulting structures.

Graph-Based

A dependency tree is a special case of a dependency graph that spawns from an artificial root and is acyclic. Because of this we can look at a large history of work in graph theory to address finding the best spanning tree for each dependency graph. The most common form of this type of dependency parsing is called arc-factored parsing and deals with the parameterization of the edge weights. The main drawback of these methods is that for non-projective trees, the worst case scenario for most methods is a complexity of $O(n^3)$ (Eisner [1996]). However, for non-projective parsing Chu-Liu-Edmond's algorithm has a complexity of $O(n^2)$ (McDonald et al. [2005]). The most common tool for doing this is MST parser, which is also used in the noun-phrase bracketing experiments described later in this paper.

Transition-Based

Transition-based parsing creates a dependency structure that is parameterized over the transitions used to create a dependency tree. This is closely related to the the shift-reduce constituency parsing algorithms. Due to the notion of picking transitions in an abstract machine, the algorithms used for these systems tend to be greedy. The benefit of this is that the algorithms have a linear time complexity. However, due to the greedy algorithms, longer arc parses can cause error propagation across each transition (Kübler et al. [2009]). The standard tool for transition-based algorithms is Malt Parser (Nivre et al. [2007b]) which in the shared tasks was often tied with the best performing systems.

Constituent Transformation

While not a true dependency parser, one technique often applied is to take a state of the art constituent parser and transform its phrase based output into dependency relations. This has been shown to also be state-of-the-art in accuracy for dependency parsing in English. This method has also been applied to the Czech language with Collin's parser. One path of research should test how this process works in other languages and for treebanks specifically annotated for dependency relations. In most cases the models are built from the Penn Treebank, a constituent based treebank (Marcus et al. [1993]), using a phrase based parser. Then to parse a sentence into a dependency structure, the phrase based output is processed with a conversion tool e.g. Penn Converter (Johansson and Nugues [2007]) or Stanford Converter (Marneffe et al. [2006]). Versions of these converters were used in the CoNLL shared task to create dependency treebanks for a variety of the languages. For my experiments I will make use of Charniak and Johnson [2005] as well as other constituent parsers.

Improving Dependency Parsing Accuracy

Current methods are aimed at improving accuracy of state of the art parsers, not at increasing the breadth of languages. Although the shared competitions have always stressed multilingual application, the final results have always been concerned with the increase in LAS and UAS in particular languages. To address this issue there are two main approaches I plan to examine: domain adaptation and annotation structure.

Domain Adaptation

One major approach to improving parsing accuracy is to better model certain domains. Later shared tasks started testing this ability in models. The idea is that a parsing model is trained on one type of text, such as financial news, and must be applied on a different domain, such as blogs. This contains syntactical, style, and lexical changes that are very difficult for many models to adjust for. Three main techniques are being researched to address domain adaptation: self training, up-training, and model selection and combination. Evaluation of domain adaptation can be tricky. Initially I will look to converting constituency treebanks of different genres into dependency treebanks for testing purposes. Although when it comes to testing in domains such as social media, new dependency structures should be created.

Self Training. Self training is the idea of training a parser on its own output. This is particularly useful when a parser was trained on one domain and you are trying to extend it to another domain. While the parser has more errors than would be normally expected, the lower accuracy output improves future outputs on the same domain when used as training data in future iterations.

Up-training. Up-training takes a slightly different approach by training a particular parser on output of different parsers. Current research has applied this to make faster parsers more accurate by up-training with the slower but more accurate parser's output. This approach has been used to change the domain of a parser as well, using the slower model to parse the out of domain data. Similar techniques could be used, but instead of basing the up-training of a higher accuracy model it could be based on a classification of smaller more domain specific models (Petrov et al. [2010]).

Model Selection and Combination. As with most NLP tools a logical path of research is to apply ensemble methods and other techniques as a way to combine the outputs of many different parsers. This, in theory, combines the benefits of each parser. There are three main approaches to this technique. First, a classifier can be used to determine which model/parser should be used for a particular sentence. Second, the outputs of all parsers can be algorithmically combined for one final output (McClosky et al. [2010]). The third way is using one model as a feature, or input, to another model. This was seen to have a positive effect when using Malt Parser as a feature to MST Parser (McDonald and Nivre). Whether through a voting schema or an algorithm similar to maximum spanning tree, one can see the usefulness of using multiple parsers to select the correct edge. We should be able to minimize errors that would be caused by syntactic or style changes caused by a domain switch.

Annotation Structure

Annotation structure and style play an important role in dependency parsing. The specificity and the structure decided upon may change the dependency parsing accuracy along with the accuracy of NLP tools that use the dependency structure output.

Noun-Phrase Bracketing. Initially in the Penn Treebank (Marcus et al. [1993]) noun phrases were treated as flat structures. For this reason most parsing systems have looked at these structures as flat. However, recently, noun phrases were annotated with their correct structure in (Vadas and Curran [2007a] Vadas and Curran [2007b] Vadas and Curran [2008] Vadas and Curran [2007c]). It is a question whether these annotations help in the training of dependency parsers and more importantly whether these structures will aid in other NLP tasks that make use of dependency structures such as machine translation. TectoMT (Žabokrtský et al. [2008]) is used as our machine translation system since it makes direct use of dependency structure in its tree transformation stages. Due to increased specificity we find a slight decrease but statistically insignificant change in parsing accuracy when using noun-phrase bracketing. However, even with a possibly lower parsing accuracy scores, the system's

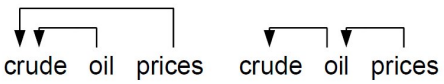

Figure 1. How noun-phrase structure can lead to ambiguities

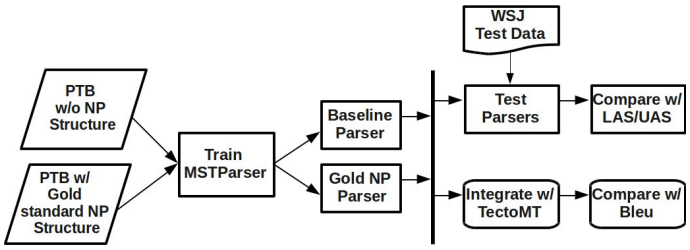

Figure 2. Flow of noun-phrase methodology and experimental decisions

Bleu scores improved with statistical significance over the baseline system, the one which didn't use noun-phrase bracketing. Both parsers were tested against their respective annotation standards.

Systems	Bleu
Baseline Parser	9.47
Gold Parser	**9.70**

Table 1. TectoMT results of a complete system run with both the Baseline Parser and Gold Parser

Both systems are tested on WMT08 data. Results are an average of 1,000 bootstrapped test sets with replacement using a pairwise comparison (Koehn [2004]), and the improvement in Bleu score is statistically significant with 95% confidence (Green [2011]).

Coordination structure. While noun phrase bracketing was a lack of structure, there are many situations where a structure is annotated but the annotators decided on a particular annotation style. For instance with coordination structure. There are standards that are left branching, right branching, coordination first, and coordination as the head of the dependency. No method is any more linguistically sound then any other method.

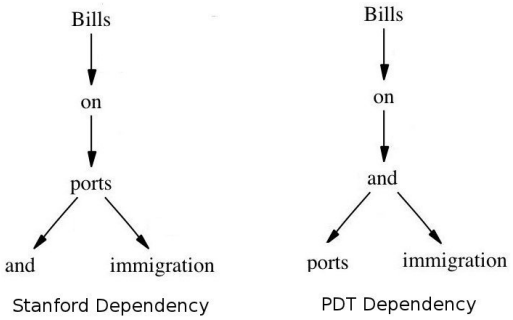

Figure 3. How Stanford dependency structure handles coordination (taken from the Stanford dependency web page)

Testing different annotation schemes for this and other structures that do not have linguistic backing behind their annotation schemes will be a good contribution to the field of dependency parsing and annotation. One style of annotation may work significantly better for a particular class of languages. Each annotation structure should be empirically tested to find the best combination for each language and machine translation pair.

Future Work and Conclusion

Increasing accuracy of the current parsers should not be the only goal going forward. There are 6,000+ languages in the world and very few of them have dependency trees available. This eliminates the possibility of any supervised training without a very heavy cost in creating the training data from scratch. For this reason, unsupervised methods should be researched and developed further. Current

techniques such as Klein and Manning [2004] only get in the 40% - 60% range in undirected and unlabeled accuracy, depending on language and treebank. This is a slight improvement over a baseline that characterizes a dependency as adjacency related. Work in this area is advantageous since the cost of manually creating dependency structures is a bottleneck for most languages in the world and unsupervised methods will give these languages a starting point into the field.

Initial research on the differences between a variety of parsers has been conducted in regards to their effect on machine translation (Popel et al. [2011]). This research should be expanded to include parser combination.

While dependency parsing has made many advancements with the shared task competitions, there is room for improvement to all current models. I hope to primarily focus on annotation standards, domain adaptation, and model combination to improve current performance. Furthermore, I hope to research and improve dependency parsing for under-resourced languages through unsupervised learning.

Acknowledgments. This research has received funding from the European Commission's 7th Framework Program (FP7) under grant agreement n° 238405 (CLARA), and from grant MSM 0021620838.

References

Buchholz, S. and Marsi, E., CoNLL-X shared task on multilingual dependency parsing, in *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pp. 149–164, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://portal.acm.org/citation.cfm?id=1596276.1596305, 2006.

Charniak, E. and Johnson, M., Coarse-to-fine n-best parsing and maxent discriminative reranking, in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pp. 173–180, Association for Computational Linguistics, Stroudsburg, PA, USA, URL http://dx.doi.org/10.3115/1219840.1219862, 2005.

Eisner, J., Three new probabilistic models for dependency parsing: An exploration, in *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pp. 340–345, Copenhagen, URL http://cs.jhu.edu/~jason/papers/#coling96, 1996.

Green, N., Effects of noun phrase bracketing in dependency parsing and machine translation, in *Proceedings of the ACL 2011 Student Session*, pp. 69–74, Association for Computational Linguistics, Portland, OR, USA, URL http://www.aclweb.org/anthology/P11-3013, 2011.

Johansson, R. and Nugues, P., Extended constituent-to-dependency conversion for English, in *Proceedings of NODALIDA 2007*, pp. 105–112, Tartu, Estonia, 2007.

Klein, D. and Manning, C., Corpus-based induction of syntactic structure: Models of dependency and constituency, in *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pp. 478–485, Barcelona, Spain, 2004.

Koehn, P., Statistical significance tests for machine translation evaluation, in *Proceedings of EMNLP 2004*, edited by D. Lin and D. Wu, pp. 388–395, Association for Computational Linguistics, Barcelona, Spain, 2004.

Kübler, S., McDonald, R., and Nivre, J., *Dependency parsing*, Synthesis lectures on human language technologies, Morgan & Claypool, US, URL http://books.google.com/books?id=k3iiup7HB9UC, 2009.

Kuhlmann, M. and Satta, G., Treebank grammar techniques for non-projective dependency parsing, in *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pp. 478–486, Association for Computational Linguistics, Athens, Greece, URL http://www.aclweb.org/anthology/E09-1055, 2009.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B., Building a large annotated corpus of english: the Penn Treebank, *Comput. Linguist.*, *19*, 313–330, URL http://portal.acm.org/citation.cfm?id=972470.972475, 1993.

Marneffe, M.-C. D., Maccartney, B., and Manning, C. D., Generating typed dependency parses from phrase structure parses, in *In LREC 2006*, 2006.

McClosky, D., Charniak, E., and Johnson, M., Automatic domain adaptation for parsing, in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 28–36, Association for Computational Linguistics, Los Angeles, California, URL http://www.aclweb.org/anthology/N10-1004, 2010.

McDonald, R. and Nivre, J., Analyzing and integrating dependency parsers, *Comput. Linguist.*, *37*, 197–230, URL http://dx.doi.org/10.1162/coli_a_00039.

McDonald, R. and Nivre, J., Characterizing the errors of data-driven dependency parsing models, in *Proceedings*

of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp. 122–131, URL http://www.aclweb.org/anthology/D/D07/D07-1013, 2007.

McDonald, R., Pereira, F., Ribarov, K., and Hajic, J., Non-projective dependency parsing using spanning tree algorithms, in *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pp. 523–530, Association for Computational Linguistics, Vancouver, British Columbia, Canada, URL http://www.aclweb.org/anthology/H/H05/H05-1066, 2005.

Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., and 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, Prague, Czech Republic, URL http://www.aclweb.org/anthology/D/D07/D07-1096, 2007a.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E., MaltParser: A language-independent system for data-driven dependency parsing, *Natural Language Engineering*, *13*, 95–135, 2007b.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J., Bleu: a method for automatic evaluation of machine translation, in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pp. 311–318, Association for Computational Linguistics, Morristown, NJ, USA, URL http://dx.doi.org/10.3115/1073083.1073135, 2002.

Petrov, S., Chang, P.-C., Ringgaard, M., and Alshawi, H., Uptraining for accurate deterministic question parsing, in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 705–713, Association for Computational Linguistics, Cambridge, MA, URL http://www.aclweb.org/anthology/D10-1069, 2010.

Popel, M., Mareček, D., Green, N., and Žabokrtský, Z., Influence of Parser Choice on Dependency-Based MT, in *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pp. 433–439, Association for Computational Linguistics, Edinburgh, Scotland, URL http://www.aclweb.org/anthology/W11-2153, 2011.

Vadas, D. and Curran, J., Adding noun phrase structure to the Penn Treebank, in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 240–247, Association for Computational Linguistics, Prague, Czech Republic, URL http://www.aclweb.org/anthology/P07-1031, 2007a.

Vadas, D. and Curran, J. R., Parsing internal noun phrase structure with Collins' models, in *Proceedings of the Australasian Language Technology Workshop 2007*, pp. 109–116, Melbourne, Australia, URL http://www.aclweb.org/anthology/U07-1016, 2007b.

Vadas, D. and Curran, J. R., Large-scale supervised models for noun phrase bracketing, in *Conference of the Pacific Association for Computational Linguistics (PACLING)*, pp. 104–112, Melbourne, Australia, URL http://www.it.usyd.edu.au/~james/pubs/pdf/pacling07bracket.pd%f, 2007c.

Vadas, D. and Curran, J. R., Parsing noun phrase structure with CCG, in *Proceedings of ACL-08: HLT*, pp. 335–343, Association for Computational Linguistics, Columbus, Ohio, URL http://www.aclweb.org/anthology/P/P08/P08-1039, 2008.

Žabokrtský, Z., Ptáček, J., and Pajas, P., TectoMT: Highly Modular MT System with Tectogrammatics Used as Transfer Layer, in *Proceedings of the 3rd Workshop on Statistical Machine Translation, ACL*, pp. 167–170, 2008.