Improving Data Driven Dependency Parsing using Clausal Information

Phani Gadde, Karan Jindal, Samar Husain, Dipti Misra Sharma, Rajeev Sangal
Language Technologies Research Centre, IIIT-Hyderabad, India.
phani.gadde@research.iiit.ac.in, karan_jindal@students.iiit.ac.in,
{samar,dipti,sangal}@mail.iiit.ac.in

Abstract

The paper describes a data driven dependency parsing approach which uses clausal information of a sentence to improve the parser performance. The clausal information is added automatically during the parsing process. We demonstrate the experiments on Hindi, a language with relatively rich case marking system and free-word-order. All the experiments are done using a modified version of MSTParser. We did all the experiments on the ICON 2009 parsing contest data. We achieved an improvement of 0.87% and 0.77% in unlabeled attachment and labeled attachment accuracies respectively over the baseline parsing accuracies.

1 Introduction

Linguistic analysis of morphologically rich free-word-order languages (MoRFWO) using dependency framework have been argued to be more effective (Shieber, 1985; Mel'čuk, 1988, Bharati et al., 1993). Not surprisingly, most parsers for such languages are dependency based (Nivre et al., 2007a; Bharati et al., 2008a; Hall et al., 2007). In spite of availability of annotated treebanks, state-of-the-art parsers for MoRFWO have not reached the performance obtained for English. Some of the reasons stated for the low performance are small treebank size, complex linguistic phenomenon, long-distance dependencies, and non-projective structures (Nivre et al., 2007a, 2007b; Bharati et al., 2008a).

Several approaches have been tried to handle these difficulties in such languages. For Hindi, Bharati et al. (2008a) and Ambati et al. (2009) used semantic features in parsing to reduce the negative impact of unavailable syntactic features and showed that use of minimal semantics can help in identifying certain core dependency labels. Various attempts have proved to simplify the structure by dividing the sentence into suitable linguistic units (Attardi and Dell'Orletta 2008; Bharati et al., 1993, 2008b, 2009; Husain et al., 2009). These approaches handle complex structures by breaking the parsing process into several steps. Attardi and Dell'Orletta (2008) used chunk information as a feature to MaltParser (Nivre et al., 2007a) for parsing English. Bharati et al., 1993 used the notion of local word groups, while Bharati et al., 2009 and Husain et al., 2009 used clauses.

In this paper, we describe a data driven dependency parsing approach which uses clausal information of a sentence to improve the parser performance. Previous attempts at data driven parsing for Hindi have failed to exploit this feature explicitly. The clausal information is added automatically during the parsing process. We demonstrate the experiments on Hindi[1]. All the experiments are done using a modified version of MSTParser (McDonald et al., 2005a and the references therein) (henceforth MST) on the ICON 2009 parsing contest[2] (Husain, 2009) data. We achieved an improvement of 0.87% and 0.77% in unlabeled attachment and labeled attachment accuracies respectively over the baseline parsing accuracies.

[1] Hindi is a verb final language with free word order and a rich case marking system. It is an official language of India and is spoken by ~800 million people.
[2] http://www.icon2009.in/contests.html

Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, pages 657–660,
Los Angeles, California, June 2010. ©2010 Association for Computational Linguistics

2 Why Clausal Information?

Traditionally, a clause is defined as a group of words having a subject and a predicate. Clause boundary identification is the process of dividing the given sentence into a set of clauses. It can be seen as a partial parsing step after chunking, in which one tries to divide the sentence into meaningful units. It is evident that most of the dependents of words in a clause appear inside the same clause; in other words the dependencies of the words in a clause are mostly localized within the clause boundary.

In the dependency parsing task, a parser has to disambiguate between several words in the sentence to find the parent/child of a particular word. This work is to see whether the clause boundary information can help the parser to reduce the search space when it is trying to find the correct parent/child for a word. The search space of the parser can be reduced by a large extent if we solve a relatively small problem of identifying the clauses. Interestingly, it has been shown recently that most of the non-projective cases in Hindi are inter-clausal (Mannem et al., 2009). Identifying clausal boundaries, therefore, should prove to be helpful in parsing non-projective structures. The same holds true for many long-distance dependencies.

3 Experimental Setup

3.1 Dataset

The experiments reported in this paper have been done on Hindi; the data was released as part of the ICON 2009 parsing contest (Husain, 2009). The sentences used for this contest are subset of the Hyderabad Dependency Treebank (HyDT) developed for Hindi (Begum et al., 2008). The dependency relations in the treebank are syntactico-semantic. The dependency tagset in the annotation scheme has around 28 relations. The dependency trees in the treebank show relations between chunk heads. Note, therefore, that the experiments and results described in this paper are based on parse trees that have chunk head as nodes.

The data provided in the task contained morphological features along with the lemma, POS tag, and coarse POS tag, for each word. These are six morphological features namely category, gender, number, person, vibhakti[3] or TAM[4] markers of the node

3.2 Clause Boundary Identifier

We used the Stage1[5] parser of Husain et al. (2009), to provide the clause boundary information that is then incorporated as features during the actual parsing process. The Stage1 parser uses MST to identify just the intra-clausal relations. To achieve this, Husain et al., introduce a special dummy node named _ROOT_ which becomes the head of the sentence. All the clauses are connected to this dummy node with a dummy relation. In effect the Stage1 parser gives only intra-clausal relations. In the current work, we used MaltParser[6] (Nivre et al., 2007b) (henceforth Malt) to do this task. This is because Malt performs better than MST in case of intra-clausal relations, which are mostly short distance dependencies. We use the same algorithm and feature setting of Bharati et al., (2008a) to train the Stage1 parser.

Since the above tool parses clauses, therefore along with the clause boundary information we also know the root of the clausal sub-tree. Several experiments were done to identify the most optimal set of clausal features available from the partial parse. The best results are obtained when the clause boundary information, along with the head information i.e. head node of a clause, is given as a feature to each node.

We trained the Stage1 parser by converting the treebank data into the stage1 format, following the steps that were given in Husain et al. (2009). This conversion depends on the definition of the clause. We experimented with different definitions of clause in order to tune the tool to give the optimal clause boundary and head information required for parsing. For the results reported in this paper, a clause is a sequence of words, with a single verb, unless the verb is a child of another verb.

[3] Vibhakti is a generic term for preposition, post-position and suffix.

[4] TAM: Tense, Aspect and Modality.

[5] Stage1 handles intra-clausal dependency relations. These relations generally correspond to the argument structure of the verb, noun-noun genitive relation, infinitive-noun relation, adjective-noun, adverb-verb relations, etc.

[6] Malt version 1.2

	Precision	Recall
Clause Boundary	84.83%	91.23%
Head Information	92.42%	99.40%

Table 1. Accuracies of the features being used

Table 1 gives the accuracy of the clausal information being used as features in parsing. It is clear from Table1 that the tool being used doesn't have very high clause boundary identification performance; nevertheless, the performance is sufficient enough to make an improvement in parsing experiments. On the other hand, the head of the clause (or, the root head in the clausal sub-tree) is identified efficiently. All the above experiments for parameter tuning were done on the development data of the ICON 2009 parsing contest.

3.3 Parser

We used MSTParser[7] for the actual parsing step. MST uses Chu-Liu-Edmonds Maximum Spanning Tree Algorithm for non-projective parsing and Eisner's algorithm for projective parsing (Eisner, 1996). It uses online large margin learning as the learning algorithm (McDonald et al., 2005b).

We modified MST so that it uses the clause boundary. Unlike the normal features that MST uses, the clause boundary features span across many words.

.

4 Experiments and Results

We experimented with different combinations of the information provided in the data (as mentioned in 3.1). Vibhakti and TAM fields gave better results than others. This is consistent with the best previous settings for Hindi parsing (Bharati et al., 2008a, Ambati et al., 2009). We used the results obtained using this setting as our baseline (F1).

We first experimented by giving only the clause inclusion (boundary) information to each node (F2). This feature should help the parser reduce its search space during parsing decisions. Then, we provided only the head and non-head information (whether that node is the head of the clause or not) (F3). The head or non-head information helps in handling complex sentences that have more than one clause and each verb in the sentence has its own argument structure. We achieved the best performance by using both as features (F4) during the parsing process.

	LA (%)	UA (%)	L (%)
F1	73.62	91.00	76.04
F2	72.66	91.00	74.74
F3	73.88	91.35	75.78
F4	74.39	91.87	76.21

Table 2. Parsing accuracies with different features

Table 2 gives the results for all the settings. It is interesting to note that the boundary information (F1) alone does not cross the baseline; however this feature is reliable enough to give the best performance when combined with F3.

5 Observations

We see from the above results (F4 in Table 2) that there is a rise of 0.87% in UA (unlabeled attachment) and 0.77% in LA (labeled attachment) over previous best (F1). This shows the positive effect of using the clausal information during the parsing process.

We analyzed the performance of both the parsers in handling the long distance dependencies and non-projective dependencies. We found that the non-projective arcs handled by F4 have a precision and recall of 41.1% and 50% respectively for UA, compared to 30.5% and 39.2% for the same arcs during F1.

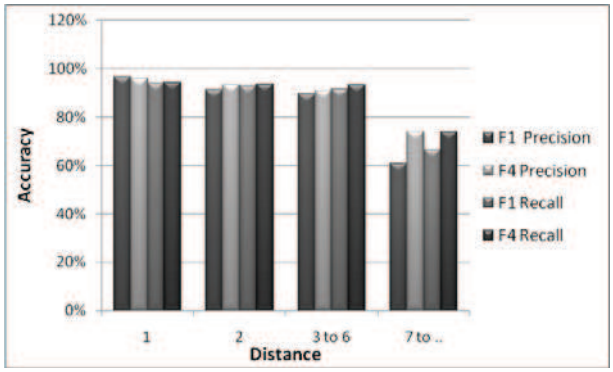

Figure 1. Distance stats

Figure 1 compares the accuracies of the dependencies at various distances. It is clear that the effect of clausal information become more

[7] MST version 0.4b

pronounced as the distance increases. This means F4 does help the parser in effectively handling long distance dependencies as well.

6 Conclusion and Future Work

The results show that there is a significant improvement in the parsing accuracy when the clausal information is being used.

The clausal information is presently being used only as attachment features in MST. Experiments can be done in future, to find out if there is a label bias to the clause boundary, which also helps in reducing the search space for specific labels. Improving the feature set for the labeled parse also improves the unlabeled attachment accuracy, as MST does attachments and labels in a single step, and the labels of processed nodes will also be taken in features.

We can see from Table1 that the precision of the clause boundary is 84.83%. Using a tool, targeted at getting just the clausal information, instead of using a parser can improve the accuracy of the clausal information, which helps improving parsing.

References

B. R. Ambati, P. Gadde, and K. Jindal. 2009. Experiments in Indian Language Dependency Parsing. In *Proceedings of the ICON09 NLP Tools Contest: Indian Language Dependency Parsing,* pp 32-37.

B. R. Ambati, P. Gade and C. GSK. 2009. Effect ofMinimal Semantics on Dependency Parsing. In the *Proceedings of RANLP 2009 Student Research Workshop*.

G. Attardi and F. Dell'Orletta. Chunking and Dependency Parsing. *LREC Workshop on Partial Parsing: Between Chunking and Deep Parsing*. Marrakech, Morocco. 2008.

R. Begum, S. Husain, A. Dhwaj, D. Sharma, L. Bai, and R. Sangal. 2008. Dependency annotation scheme for Indian languages. *In Proceedings of IJCNLP-2008.*

A. Bharati and R. Sangal. 1993. Parsing Free Word Order Languages in the Paninian Framework. *Proceedings of ACL:93*.

A. Bharati, S. Husain, B. Ambati, S. Jain, D. Sharma and R. Sangal. 2008a. Two Semantic features make all the difference in Parsing accuracy. In *Proceedings. of International Conference on Natural Language Processing-2008.*

A. Bharati, S. Husain, D. Sharma, and R. Sangal. 2008b. A two stage constraint based dependency parser for free word order languages. *In Proceedings. of COLIPS International Conference on Asian Language Processing. Thailand. 2008.*

A. Bharati, S. Husain, D. M. Sharma and R. Sangal. Two stage constraint based hybrid approach to free word order language dependency parsing. In the *Proceedings of the 11th International Conference on Parsing Technologies (IWPT09). Paris. 2009.*

J. Hall, J. Nilsson, J. Nivre, G. Eryigit, B. Megyesi, M. Nilsson,M. Saers.2007. Single Malt or Blended? A Study in Multilingual Parser Optimization. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*.

S. Husain. 2009. Dependency Parsers for Indian Languages. In *Proceedings of ICON09 NLP Tools Contest:Indian Language Dependency Parsing. Hyderabad, India. 2009.*

S. Husain, P. Gadde, B. Ambati, D. M. Sharma and Rajeev Sangal. 2009. A modular cascaded approach to complete parsing. In the *Proceedings of COLIPS International Conference on Asian Language Processing. Singapore. 2009.*

P. Mannem and H. Chaudhry.2009. Insights into Non-projectivity in Hindi. In *ACL-IJCNLP Student paper workshop. 2009.*

R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. 2005a. Non-projective dependency parsing using spanning tree algorithms. In the *Proceedings of HLT/EMNLP*, pp. 523–530.

R. McDonald, K. Crammer, and F. Pereira. 2005b. Online large-margin training of dependency parsers. In the *Proceedings of ACL 2005.* pp. 91–98.

I. A. Mel'Cuk. 1988. Dependency *Syntax: Theory and Practice*, State University Press of New York.

J. Nivre, J. Hall, S. Kubler, R. McDonald, J. Nilsson, S. Riedel and D. Yuret. 2007a. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*.

J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov and E Marsi. 2007b. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2), 95-135.

S. M. Shieber. 1985. Evidence against the context-freeness of natural language. In *Linguistics and Philosophy*, p. 8, 334–343.