Vol. 1, Issue 4, pp.1751-1754

Dependency Parsing for Telugu

Mrs.M.Humera Khanam, Mr.Palli Suryachandra, Prof.K.V.Madhu Murthy

Department of Computer Science,SVUCE,SV University, Tirupathi,India)
Department of Computer Science,SVUCE,SV University, Tirupathi,India)
Department of Computer Science,SVUCE,SV University, Tirupathi,India)

Abstract
In this paper we present our experiments in parsing for Telugu language. We explore two data driven parsers Malt and MST and compare the results of both the parsers. We describe the data and parser settings used in detail. Some of these are specific to either one particular or all the Indian Languages. The average of best unlabeled attachment, labeled attachment and labeled accuracies are 88.43%, 69.71% and 70.01% respectively .We are also presented which parser gives best results for different sentence types in Telugu.

Keywords – Telugu, Malt, MST

1.INTRODUCTION

Parsing is one of the major tasks which helps in understanding the natural language. It is useful in several natural language applications. Machine translation, anaphora resolution, word sense disambiguation, question answering, summarization are few of them. This led to the development of grammar-driven, data-driven and hybrid parsers. Due to the availability of annotated corpora in recent years, data driven parsing has achieved considerable success. The availability of phrase structure Treebank's for English has seen the development of many efficient parsers. Telugu Language is morphologically rich free word order language. It has been suggested that free word order language can be handled better using the dependency based framework than the constituency based one (Hudson, 1984;Shieber, 1985; Mel'čuk, 1988,Bharati et al.,1995).As a result, dependency annotation using paninian framework is started for telugu Language (Begum et al., 2008). There have been some previous attempts at parsing Hindi following a constraint based approach (Bharati et al.1993, 2002, 2008b).Due to availability of tree-bank for Hindi, some attempts are made at building statistical (Bharati et al., 2008a, Husain et al.,2009; Ambati et al., 2009) and hybrid parsers (Bharati et al., 2009). In all these approaches both syntactic and semantic cues are explored to reduce the confusion between ambiguous dependency tags. In this paper we describe our experiments in parsing Telugu in detail. Some of these are specific to either one particular or all the Indian Languages and some in general to any kind of language. We explore two data-driven parsers Malt and MST and compare results of both the parsers. The average of best unlabeled attachment, labeled attachment and labeled accuracies are

90.52%, 67.93% and 69.25% for malt and 89.65%,64.99% and 65.17% for MST parsers respectively. The paper is arranged as follows, in section 2, we present general information about data,we describe our approach for parsing. Section 3 describes the data and parser settings for Telugu language. We present our results in section 4. We conclude our paper in section 5.

2. Approch

Malt Parser:

Malt Parser (Nivre et al., 2006) implements which has two essential components:

•A transition system for mapping sentences into dependency trees

•A classifier for predicting the next transition for every possible system configuration

Transition Systems:

MaltParser comes with a number of built-in transition systems, but we limit our attention to the two systems that have been used in the parsing experiments: the arc-eager projective system first described in Nivre (2003) and the non-projective transition system based on the method described by Covington (2001). For a more detailed analysis of this and other transition systems for dependency parsing, see Nivre (2008).A configuration in the arc-eager projective system contains a stack holding partially processed tokens, an input buffer containing the remaining tokens, and a set of arcs representing the partially built dependency tree. There are four possible transitions (where top is the token on top of the stack and next is the next token in the input buffer):

•LEFT-ARC (r): Add an arc labeled r from next to top; pop the stack.

•RIGHT-ARC (r): Add an arc labeled r from top to next; push next onto the stack.

•REDUCE: Pop the stack.

•SHIFT: Push next onto the stack. support

We performed our experiments on malt parser version 1.4.1. Malt parser provides options for Arc-Standard, Arc-Eager, Covington Projective, Cov-ington Non-Projective, Planar and Stack parsing algorithms. It also provides options for LIBSVM and LIBLINEAR learning algorithms. We experimented with different combinations of these algorithms to arrive at the best settings.

Features, Feature selection and Templates:

An extensive list of features is prepared which we thought are appropriate in helping to form a better parse. Best settings for each data set are selected with a simple forward selector. The simple forward selector runs by appending the feature to template file one by one seeing when it meets the given criteria. A set of algorithms and classifiers are to be provided prior to the forward selector. We include the corresponding feature in the template file if the LAS increase and UAS does not decrease. We have used LIBLINEAR and LIBSVM classifiers and found out the LIBLINEAR has achieved slightly better accuracies over the LIBSVM. This is because of the new version of Malt parser has updated the LIBSVM and LIBLINEAR packages. A 5 fold cross validation has been done for selecting the best template, best algorithm and best classifier for Telugu Language data. From the set of morphological information provided along with the data, vibhakti and tam has helped in improving accuracies (Bharati et al., 2008; Ambati et al., 2009, 2010a).

MST Parser:

The parser should work with Java 1.4 and 1.5.
This is the parser described in the following papers
-Multilingual Dependency Parsing with a Two-Stage Discriminative Parser
-Online Learning of Approximate Dependency Parsing Algorithms
-Non-projective Dependency Parsing using Spanning Tree Algorithms
-Online Large-Margin Training of Dependency Parsers

Telugu is a Dravidian language which is agglutinative in nature. We have taken Telugu annotated data,which contains sentences1651words7920Unique words2964chunks5983

We used two data driven parsers Malt (Nivre et al., 2007a), and MST (McDonald et al., 2005b) for our experiments. Malt is a classifier based Shift/Reduce parser. It uses arc-eager, arc-standard, covington , projec-tive and convington non-projective algorithms for parsing (Nivre,2006). History-based feature models are used for predicting the next parser action (Black et al.,1992). Support vector machines are used for mapping histories to parser actions (Kudo and Matsumoto,2002). It uses graph transformation to handle non-projective trees (Nivre and Nilsson, 2005).MST uses Chu-Liu-Edmonds (Chu and Liu,1965; Edmonds, 1967) 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., 2005a).Malt provides an xml file, where we can specify the features for the parser. But for MST, these features are hard coded. Accuracy of the labeler of MST is very low. We tried to modify the code but couldn't get better results.

3.Settings:

Input Data:

Both the parsers take CoNLL format as input. So, we have taken data in CoNLL format for our experiments. The FEATS column of each node in the data has 6 fields. These are six morphological features namely category, gender, number, person, vibhakti5 or TAM6 markers of the node. We experimented considering different combinations of these fields for both the parsers. For Telugu language vibhakti and TAM fields gave better results than others. This is similar to the settings of Bharati et al. (2008a). They showed that for Hindi, vibhakti and TAM markers help in dependency parsing where as gender,number, person markers won't.

Malt Parser Settings:

Malt provides options for four parsing algorithms arc-eager, arc-standard, covington projective,covington non-projective. We experimented with all the algorithms for all the three languages for both the tagsets. Tuning the SVM model was difficult; we tried various parameters but could not find any fixed pattern. Finally, we tested the performance by adapting the CoNLL shared task 2007 (2007b) settings used by the same parser for various languages (Hall et. al, 2007). For feature model also after exploring general useful features, we experimented taking different combinations of the settings used in CoNLL shared task 2007 for various languages. For Telugu language,the following are the best settings.

For Parsing	Arc eager
For Learning	Liblinear

MST Parser Settings:

MST Parser is a non-projective dependency parser that searches for maximum spanning trees over directed graphs. Models of dependency structure are based on large-margin discriminative training methods. Projective parsing is also supported. MST parser contains the parameters train, train-file, model-name, training-iterations, decode-type, training-k, loss-type, order. By doing many experiments, for the following values, the parser gives best accuracies.

Training-k	6
Decode type	proj
Order	1

4.Experiments and Results:

We merged both the training and development data and did 5-fold cross-validation for tuning the parsers. We extracted best settings from the cross validation experiments. These settings are applied on the test data. Size of the test data is 150 sentences.

Results on Test Data:

Malt parser:

UAS	90.52
LAS	67.93
LA	69.25

MST Parser :

UAS	89.56
LAS	65.23
LA	66.25

Evalution for sentences :

We broadly divided sentences in Telugu as the following categories with our linguistic knowledge in Telugu. We have given examples for each sentence types.

Simple Sentences : Simple sentences contain no conjunction

Ex : nEnu ikkadaku vachAnu.

Compound Sentences : Compound sentences contain two statements that are connected by a conjunction

Ex : athanu vachAdu mariyu tirigi vellAdu.

Complex Sentences :Complex sentences contain a dependent clause and at least one independent clause.

Ex : eEme, nA kUthauru,pEru ramya.

Compound - Complex Sentences : Compound - complex sentences contain at least one dependent clause and more than one independent clause.

Ex:ramu,pOyina nela ikadaku vachi,bahumathi

tEsukunnadu mariyu selavu pI vellAdu.

We have classified our sentences based on the sentence types. We have given these classified sentences for testing to the two parsers. For simple sentences, both parsers had given good results, but for other sentence types they have shown less accuracies. We presented our results below.

Sentence type	Malt	MST
Simple	81.14	76.23
Compound	63.45	59.23
Complex	58.94	54.63
Compound-Complex	53.87	49.23

we found that both parsers were showing less accuracies for complex, compound-complex type sentences because they have long sentences, including many punctuations like comma, period, and other symbols. These punctuation symbols increases the complexity for giving best accuracies for the parsers.

5.Conclusions and Future Directions :

For Telugu language, Malt performed better over MST. We have modified the implementation of MST to handle vibhakti and TAM markers for labeling. We observed that even during unlabeled parsing some features which might not be useful in parsing are being used. We would like to modify the implementation to do experiments with features for unlabeling also. For getting best best accuracies for long sentences, we need to divide the sentences as phrases by using punctuations, which are present in the sentences. It also involves more linguistic knowledge in Telugu grammar. This is the future work we need to work for getting best accuracies.

References:

R. Begum, S. Husain, A. Dhwaj, D. M. Sharma, L.Bai, and R. Sangal. 2008. Dependency annotation scheme for Indian languages. In Proceedings of IJCNLP-2008. http://www.iiit.net/techreports/2007_78.pdf

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

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

Ryan McDonald , Kevin Lerman , Fernando Pereira.Multilingual Dependency Analysis with a Two-Stage Discriminative Parser.

Ryan McDonald , Fernando Pereira .Online Learning of Approximate Dependency Parsing Algorithms.

Ryan McDonald , Fernando Pereira , Kiril Ribarov , Jan Haji? .Non-projective Dependency Parsing using Spanning Tree Algorithms.

Ryan McDonald , Koby Crammer , Fernando Pereira .Online Large-Margin Training of Dependency Parsers.

J. Nivre and J. Nilsson. 2005. Pseudo-projective dependency parsing. In Proc. of ACL-2005, pages 99–106.

J. Nivre. 2006. Inductive Dependency Parsing. Springer.

J. Nivre and J. Hall and S. Kubler and R. McDonald and J. Nilsson and S. Riedel and D. Yuret. 2007b.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. 2007a. Malt Parser: A language-independent system for data-driven dependency parsing. Natural Language Engineering, 13(2), 95-135.

J. Nivre and R. McDonald. 2008. Integrating Graph Based and Transition-Based Dependency Parsers.In Proc. Of ACL-2008.

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

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

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

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

T. Kudo and Y. Matsumoto. 2002. Japanese dependency analysis using cascaded chunking. In CoNLL-2002. pp. 63–69.

R. Hudson. 1984. Word Grammar, Basil Blackwell,108 Cowley Rd, Oxford, OX4 1JF, England.

J. Hall, J. Nilsson, J. Nivre, G. Eryigit, B. Megyesi,M. Nilsson and 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, 933—939.

J. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In Proceedings of COLING-96, pages 340–345.

J. Edmonds. 1967. Optimum branchings. Journal of Research of the National Bureau of Standards,71B:233–240.

Y.J. Chu and T.H. Liu. 1965. On the shortest arborescence of a directed graph. Science Sinica, 14:1396–1400.

Q. Dai, E. Chen, and L. Shi. 2009. An iterative approach for joint dependency parsing and semantic role labeling. In Proceedings of the 13th Conference on Computational Natural Language Learning(CoNLL-2009), June 4-5, Boulder, Colorado,USA.June 4-5.

E. Black, F. Jelinek, J. D. Lafferty, D.M.Magerman,R. L.Mercer, and S. Roukos. 1992. Towards history-based grammars: Using richer models for probabilistic parsing. In Proc. of the 5th DARPA Speech and Natural Language Workshop, pages 31–37.

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

A. Bharati, V. Chaitanya and R. Sangal. 1995. Natural Language Processing: A Paninian Perspective,Prentice-Hall of India, New Delhi, pp. 65-106.

A. Bharati, R. Sangal, T. P. Reddy. 2002. A Constraint Based Parser Using Integer Programming,In Proceedings of ICON, 2002.www.iiit.net/techreports/2002_3.pdf

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