IJCLA VOL. 3, NO. 2, JUL-DEC 2012, PP. 123–134
RECEIVED 31/10/11 ACCEPTED 09/12/11 FINAL 08/06/12

Exploring Self-training and Co-training for Dependency Parsing

RAHUL GOUTAM AND BHARAT RAM AMBATI

IIIT-H, India

ABSTRACT

We explore the effect of self-training and co-training on Hindi dependency parsing. We use Malt parser, which is a state-of-the-art Hindi dependency parser, and apply self-training using a large unannotated corpus. For co-training, we use MST parser with comparable accuracy to the Malt parser. Experiments are performed using two types of raw corpora—one from the same domain as the test data and another, which is out-of-domain from the test data. Through these experiments, we compare the impact of self-training and co-training on Hindi dependency parsing.

KEYWORDS**:** *Bootstrapping, dependency parsing, syntax, morphologically rich language.*

1 INTRODUCTION

Parsing morphologically rich free-word-order languages like Czech, Hindi, Turkish, etc., is a challenging task. Unlike English, most of the parsers for such languages have adopted the dependency grammatical framework. It is a well-known fact that for these languages, dependency framework is better suited [18, 12, 2]. Due to the availability of annotated corpora in recent years, data driven dependency parsing has achieved considerable success. In spite of availability of annotated treebanks, state-of-the art parsers for these

languages have not reached the performance obtained for English [14]. Frequently stated reasons for low performance are small treebank size, complex linguistic phenomena, long distance dependencies, and non-projective structures [14, 15, 3].

In this paper, we try to address the problem of small treebank size. We have lots of unannotated data. One way to increase treebank size is to manually annotate this data. But it is a very time consuming task. Another way is to parse this data using an existing parser and consider these automatic parses. But, what criteria should be used for extracting reliable parses from the automatically parsed data is a really challenging task.

In this paper, we explore the effect of two bootstrapping techniques, namely, self-training and co-training and see its impact on dependency parsing accuracy. We use Malt parser, that is the state-of-the-art Hindi dependency parser, and apply self-training using a large unannotated corpus. We also use MST parser with accuracy comparable to Malt parser and apply co-training.

We use two types of unannotated corpora, one from the same domain as the test data and another from a different domain, to explore the impact of domain of unannotated data on parsing accuracy. Though we work and present our results on Hindi, this approach can be applied to other languages with small treebanks like Telugu and Bangla.

This paper is arranged as follows. In Section 2, we describe the related work in bootstrapping in parsing. In Section 3, we present the state-of-the art Hindi dependency parser. In section 4, we report our experiments and analyze the results. We conclude with possible future work in Section 5.

2 RELATED WORK

In this section, we briefly describe the major works on bootstrapping in statistical dependency parsing.

The authors of [19] perform experiments to show that unannotated data can be used to improve the performance of statistical parsers by bootstrapping techniques. The focus of their paper is on co-training between two statistical parsers but they also perform self-training experiments with each of the two parsers. Although the results of self-training are not very encouraging, co-training experiments report modest improvement in parsing accuracy. They also perform cross-genre experiments to show that co-training is beneficial even when the seed data is from a different domain compared to the unannotated data.

The authors of [17] also perform self-training by using unannotated data from two different corpora - one in-domain and the other out-of-domain. They show that parser adaptability can be enhanced via self-training. They also report significant reduction in annotation cost and amount of work because a small manually annotated seed data is used.

The authors of [10] use a two phase parser-reranker system for self-training using readily available unannotated data. The two-phase parser reranker system consists of a generative parser and a discriminative reranker. They apply self-training on the generative parser only and not on the discriminative reranker and report significant improvement in accuracy over the previous state-of-the-art accuracy for Wall Street Journal parsing.

All the above mentioned works are on phrase structure parsing of English. There is an attempt at exploring usefulness of large raw corpus for dependency parsing by [5]. They could achieve considerable improvement over baseline for Chinese using only high confident edges instead of entire sentences. In our work the focus is dependency parsing of Hindi using a discriminative parser. We also explore how domain of data affects the parser performance.

3 HINDI DEPENDENCY PARSING

In ICON 2009 and 2010, two tools contests were held that focused on Indian Language dependency parsing [6, 7]. In these contests, rule-based, constraint based, statistical and hybrid approaches were explored towards building dependency parsers for Hindi. In 2009 contest, given the gold standard chunk heads, the task was to find dependencies between them. But in 2010 contest, given words with gold features like part-of-speech (POS) and morph information, the task was to find word level dependency parse. The ICON 2010 tools contest Hindi data consists of 2972, 543 and 321 sentences for training, development and testing with an average sentence length of 22.6. This data is a part of a larger treebank [4] which is under development. This is a news corpus taken from well-known Hindi news daily.

3.1 *Baseline (State-of-the-art) System*

We consider the best system [8] in ICON 2010 tools contest as the starting point. [8] used MaltParser [15] and achieved 94.5% Unlabeled

Attachment Score (UAS) and 88.6% Labeled Attachment Score (LAS). They could achieve this using liblinear learner and nivrestandard parsing algorithm. But, as mentioned above, POS and other features used in this system were gold standard. The only available system which uses automatically extracted features and does complete word level parsing for Hindi is [1]. Though both [1] and [8] used MaltParser, the data used is the subset of the one used by the latter and the parser settings were slightly different.

Table 1. Comparison of Different Systems

System	UAS	LAS	LS
Ambati et al. (2010)+ automatic features	85.5%	75.4%	78.9%
Kosaraju et al.(2010) + gold features	94.5%	88.6%	90.0%
Kosaraju et al.(2010) + automaticFeatures	86.5%	77.9%	81.7%

Taking training data and parser settings of Kosaraju et al. (2010) and automatic features similar to Ambati et al. (2010), we developed a parser and evaluated it on the ICON 2010 tools contest test data. We could achieve LAS of 77.9% and UAS of 86.5% on test set. This is the state-of-the-art system for Hindi dependency parsing using automatic features. We consider this system as our baseline and try to explore bootstrapping techniques to improve accuracy.

4 Experiments and Analysis

4.1 *Self-Training*

The parser used for self-training experiments is the Malt parser. We apply the settings of [8] along with automatic features (last line of Table 1). The parser is first trained on the ICON 2010 training data for Hindi. The model generated is then used to parse the unannotated corpus.

In the self-training experiments, we add the data incrementally in iterations. At each iteration, 1000 sentences are chosen randomly from the unannotated corpus which has been parsed by the model generated above and added to the training data. The parser is then trained again and the generated model is used to parse the test data.

Self-training experiments were performed using two types of data: one from the news domain (in-domain) and another from a different domain comprising mostly tourism data (out-of-domain).

4.1.1 *Self-Training: In-Domain*

We have taken unannotated news corpus of about 108,000 sentences. As a first step, we have cleaned the data. In this process, we removed the repeated sentences, and very large sentences (greater than 100 words per sentence).

Performance of the system on test data for the first 50 iterations is shown in Figures 1 and 2. Best accuracies of 78.6% LAS and 87% UAS were achieved, an improvement of 0.7% and 0.5% respectively.

4.1.2 *Self-Training: Out-of-Domain*

In this experiment, unannotated data from a domain different from the actual training and testing data is used for self- training. For this purpose, we have taken a non-news corpus of about 700,000 sentences. Similar to in-domain data, we first cleaned the data.

Performance of the resulting system on test data for the first 50 iterations is shown in Figures 1 and 2. There isnt any improvement in LAS over the baseline. Best accuracy of 77.8% LAS and 86.8% UAS was observed, an improvement of 0.3% in UAS, but a decrement of 0.1% in LAS.

4.2 *Co-Training*

The parsers used for co-training experiments are the Malt parser and the MST parser [6]. We have optimized the MST parser by modifying the feature extraction module so that the parser extracts relevant features for a morphologically rich language like Hindi. The best accuracy we achieved on the test set is 77.0% LAS and 86.5% UAS.

Using the best settings of the MST parser obtained above, a model is trained using the training set of ICON 2010 Hindi data with automatic features. This model is then used to parse the unannotated data. As in the self-training experiments, data are added incrementally in iterations. At each iteration, 1000 sentences are chosen randomly from the MST parser output and added to the training data of Malt parser. Malt parser is then trained again and the generated model is used to parse the test data.

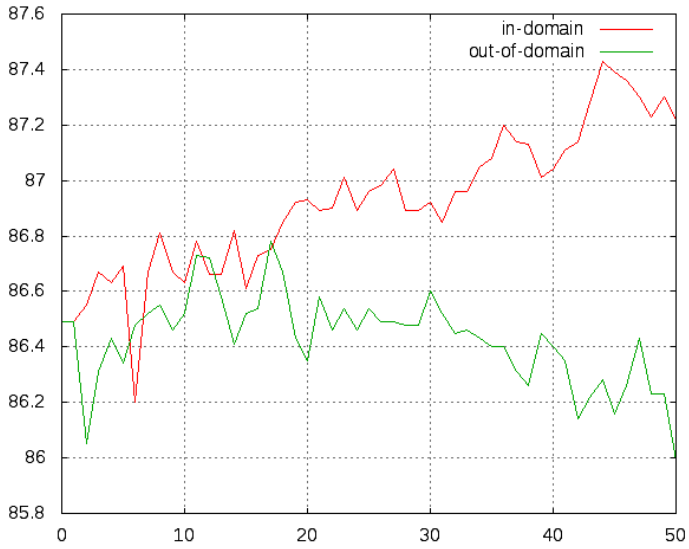

Fig. 1. Self-Training: UAS

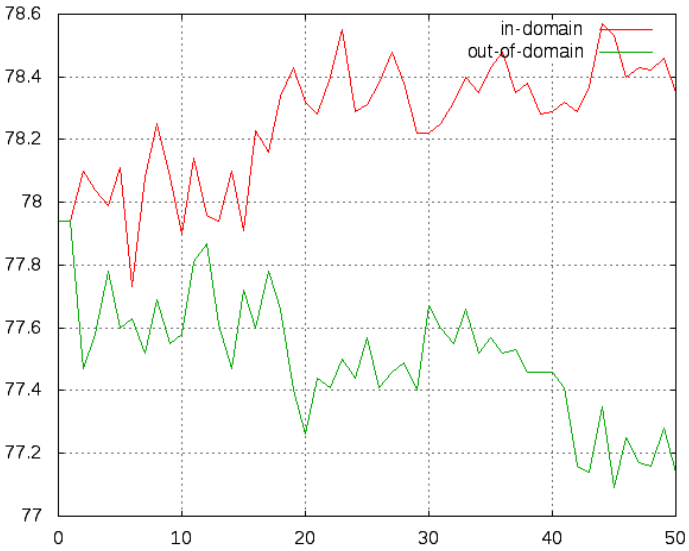

Fig. 2. Self-Training: LAS

4.2.1 *Co-Training: In-Domain*

The unannotated corpus used is the same as that used in self-training: in-domain experiments. Performance of the system is shown in Figures 3 and 4. Best accuracy of 78.6% LAS and 87.0% UAS was achieved, an improvement of 0.7% and 0.5%, respectively.

4.2.2 *Co-Training: Out-of-Domain*

The corpus used for out-of-domain experiments is the same as that used in self-training: out-of-domain experiments. Performance of the system is shown in Figures 3 and 4. There is a decrease in both UAS and LAS. The decrease in LAS is more compared to UAS.

4.3 *Co-Training: Sentence Selection via Agreement*

In this experiment, Malt and MST parsers are first trained using the training set of ICON 2010 Hindi data and then used to parse the unannotated data. The output of both parsers are then compared and sentences for which both Malt and MST parsers give the same parse are selected for bootstrapping. As in previous experiments, data is added incrementally with 1000 sentences per iteration. The 1000 sentences are chosen randomly from the pool of selected sentences and added to the training data of Malt parser. The parser is then trained again and the generated model is used to parse the test data.

4.3.1 *In-Domain Scenario*

The unannotated news corpus has approximately 108,000 sentences and both Malt and MST parsers gave the same parse for 10,461 sentences. These 10,461 sentences constitute our pool of selected sentences.

Performance of the system is shown in Figures 5 and 6. We achieved 78.8% LAS and 87.1% UAS, an improvement of 0.9% and 0.6% respectively over the baseline.

4.3.2 *Out-of-Domain Scenario*

The unannotated non-news corpus has approximately 700,000 sentences and both Malt and MST parsers gave the same parse for 45,328 sentences. These 45,328 sentences constitute our pool of selected sentences.

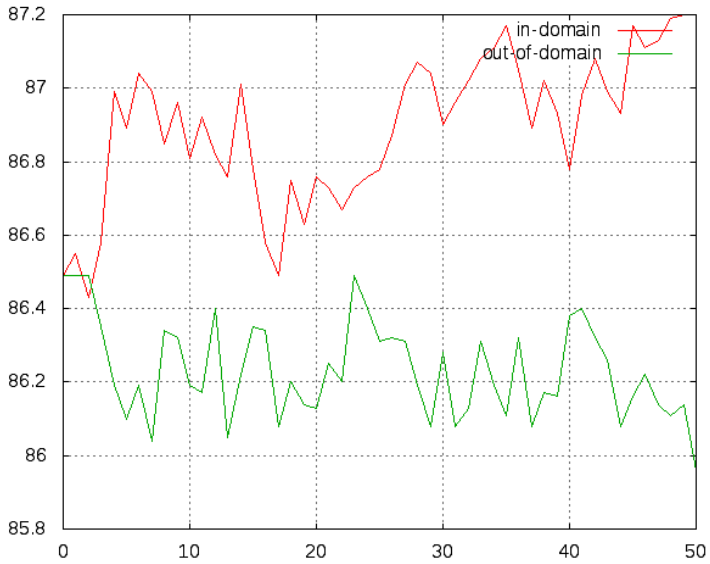

Fig. 3. Co-Training : UAS

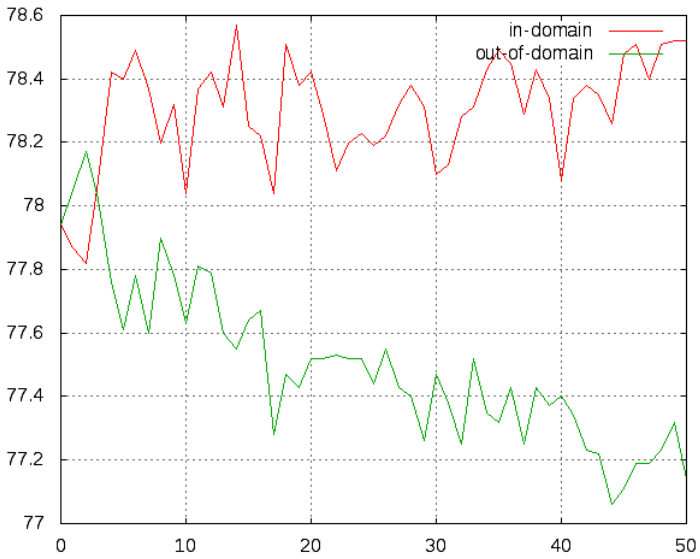

Fig. 4. Co-Training : LAS

Performance of the system for the first 12 iterations is shown in Figures 5 and 6. The remaining iterations are not shown because they follow a similar trend as the first few iterations. There is no improvement in LAS and UAS.

4.4. *Analysis*

Table 2 gives the summary comparing all the experiments performed. The * mark in the table shows that accuracy is statistically significant over the baseline. Significance is calculated using McNemar's test ($p \leq 0.05$) made available with MaltEval [13].

Table 2. Summary of experiments

System	UAS	LAS	LS
Baseline	86.5%	77.9%	81.7%
In-Domain Self-Training	87.0%*	78.6%*	82.3%*
Out-of-Domain Self-Training	86.8%	77.8%	81.6%
In-Domain Co-Training	87.0%*	78.6%*	82.2%*
Out-of-Domain Co-Training	86.5%	78.2%	82.0%
In-Domain Co-Training via Agreement	87.1%*	78.8%*	82.6%*
Out-of-Domain Co-Training v/ Agreement	86.5%	77.8%	81.6%

We could achieve significant improvement in accuracy over state-of-the-art system by applying bootstrapping with unannotated data from the same domain. There was a decrease in parser performance when data from a different domain was used. This clearly showed the importance of domain when applying bootstrapping in statistical parsers. Self-training and co-training both gave roughly the same improvement in performance for both UAS and LAS which is achieved after 23 iterations for self-training and 14 iterations for co-training. Co-training via agreement gave greater improvement in less number of iterations due to better sentence selection criteria.

We have also experimented with different sentence selection criteria. Classification scores were obtained for each labeled attachment for both the Malt and MST parsers. These scores represent the liblinear classification score for Malt and the maxent labeler probability for MST. These scores were then used to calculate the confidence score of a sentence.

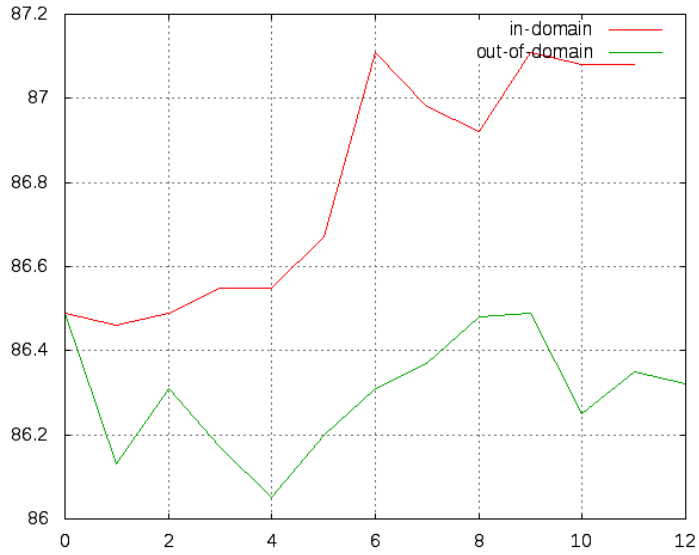

Fig. 5. Co-training v/ agreement: UAS

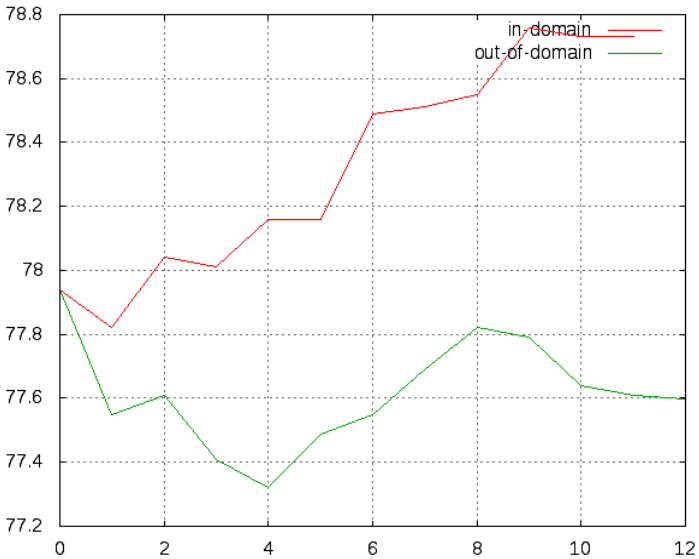

Fig. 6. Co-training v/ agreement: LAS

We have experimented with different methods to calculate confidence score of sentence, such as

- average score of labeled attachment,
- threshold on maximum and minimum score of all labeled attachments in sentence,
- normalized product,
- considering inter-chunk attachment scores only as accuracy of intra-chunk attachment is very high [1].

The most confident sentences were then added to the training data for the next iteration of bootstrapping. All these methods gave modest improvement, but the best improvement we could obtain was by selecting sentences via agreement between the two parsers.

We analyzed the label-wise precision of in-domain self-training experiments and found that there is significant improvement in precision of labels for which Malt parser is poor at identifying. For example, precision of label “main” (root of the sentence) increased from 65.4% to 84.8%. We observed two major reasons for it:

Increase in vocabulary. Approximately 30% of nodes correctly classified as “main” in the self-trained system (but not in the baseline system) are out-of-vocabulary words.

Most of the remaining cases were highly ambiguous that got correctly identified because of better feature tuning. In case of co-training, improvement in recall was observed across most labels, but there was a drop in precision.

5 Conclusion and Future Work

We explored the effect of applying bootstrapping techniques self-training and co-training on Hindi Dependency Parsing. We also performed in-domain and out-of-domain experiments to analyze the impact of domain on bootstrapping. We also explored different selection criteria and our results showed that the selection criteria need not be very sophisticated. Even random selection of sentences or simple agreement between the two parsers for sentence selection gives significant improvement in parsing accuracy.

In the future, instead of using whole sentence parse, we plan to use sub-parses that the parser is confident about to be used in

bootstrapping. We also plan to apply bootstrapping in other Indian languages such as Telugu and Bangla.

REFERENCES

1. Ambati, B. R., Gupta, M., Husain, S., Sharma, D. M.: 2010. A high recall error identification tool for Hindi treebank validation. Proceedings of The7th International Conference on Language Resources and Evaluation (LREC), Valleta, Malta.
2. Bharati, A., Chaitanya, V., Sangal, R.: 1995. Natural Language Processing: A Paninian Perspective. Prentice-Hall of India, New Delhi.
3. Bharati, A Husain, S., Ambati, B., Jain, S., Sharma, D., Sangal, R.: 2008. Two semantic features make all the difference in parsing accuracy. In Proceedings of ICON-08.
4. Bhatt, R., Narasimhan, B., Palmer, M., Rambow, O., Sharma, D. M., Xia, F. 2009.: Multi Representational and Multi-Layered Treebank for Hindi/Urdu. In proceedings of the Third Linguistic Annotation Workshop at 47 th ACL and 4th IJCNLP.
5. Chen, W., Wu, Y., Isahara, H.: 2008. Learning reliable information for dependency parsing adaptation. In Proceedings of the 22nd International Conference on Computational Linguistics (COLING).
6. Husain, S.: 2009. Dependency Parsers for Indian Languages. In Proceedings of ICON09 NLP Tools Contest: Indian Language Dependency Parsing. Hyderabad, India.
7. Husain, S., Mannem, P., Ambati, B., Gadde, P.: 2010. The ICON- 2010 Tools Contest on Indian Language Dependency Parsing. In Proceedings of ICON-2010 Tools Contest on Indian Language Dependency Parsing.Kharagpur, India.
8. Kosaraju, P., Kesidi, S. R., Ainavolu, V. B. R., Kukkadapu, P.: 2010. Experiments on Indian Language Dependency Parsing. In Proc. of the ICON-2010 NLP Tools Contest: Indian Language Dependency Parsing.
9. Mannem, P., Dara, A.: 2011. Partial Parsing from Bitext Projections. In Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics.
10. McClosky, D., Charniak, E., Johnson, M.: 2006. Effective Self- Training for Parsing. In Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics.
11. McDonald, R., Lerman, K., Pereira, F.: 2006. Multilingual dependency analysis with a two-stage discriminative parser. In Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X), pp. 216220.
12. Mel'čuk, I. A.: 1988. Dependency Syntax: Theory and Practice. State University Press of New York.

13. Nilsson, I. J., Nivre, J. 2008. Malteval: An evaluation and visualization tool for dependency parsing. In Proceedings of the Sixth LREC, Marrakech, Morocco.
14. Nivre, J., Hall, J., Kubler, S., McDonald, R., Nilsson, J., Riedel, S., Yuret, D. 2007a. TheCoNLL 2007 Shared Task on Dependency Parsing. In Proceedings of EMNLP/CoNLL-2007.
15. Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kbler, S., Marinov, S., Marsi, E. 2007b. MaltParser: A language-independent system for data-driven dependency parsing. Natural Language Engineering, 13(2), 95-135.
16. Nivre, J., Rimell, L., McDonald, R., GmezRodrguez, C. 2010. Evaluation of Dependency Parsers on Unbounded Dependencies.In proceedings of the International Conference on Computational Linguistics (COLING).
17. Reichart, R., Rappoport, A. 2007. Self-Training for Enhancement and Domain Adaptation of Statistical Parsers Trained on Small Datasets. In Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics.
18. Shieber, S. M. 1985. Evidence against the contextfreeness of natural language. In Linguistics and Philosophy, p. 8, 334343.
19. Steedman, M. Osborne, M., Sarkar, A., Clark, S., Hwa, R., Hockenmaier, J., Ruhlen, P., Baker, S., Crim, J. 2003. Bootstrapping Statistical Parsers from Small Datasets.In Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics Volume 1.
20. Aho, A. V., Ullman, J. D. 1972. The Theory of Parsing, Translation and Compiling, volume 1. Prentice-Hall, Englewood Cliffs, NJ.
21. American Psychological Association. 1983. Publications Manual. American Psychological Association, Washington, DC.
22. Association for Computing Machinery. 1983. Computing Reviews, 24(11):503–512.
23. Chandra, A. K., Kozen, D. C., Stockmeyer, L. J. 1981. Alternation. Journal of the Association for Computing Machinery, 28(1):114–133.
24. Gusfield, D. 1997. Algorithms on Strings, Trees and Sequences. Cambridge University Press, Cambridge, UK.

RAHUL GOUTAM
LANGUAGE TECHNOLOGIES RESEARCH INSTITUTE, IIIT-H,
OBH-208, GACHIBOWLI, HYDERABAD, 500032, INDIA.
E-MAIL: <RAHUL.GOUTAM@RESEARCH.IIIT.AC.IN>

BHARAT RAM AMBATI
LANGUAGE TECHNOLOGIES RESEARCH INSTITUTE, IIIT-H,
OBH-208, GACHIBOWLI, HYDERABAD, 500032, INDIA.
E-MAIL: <AMBATI@RESEARCH.IIIT.AC.IN>