Effective Parsing for Human Aided NLP Systems

Naman Jain **Sambhav Jain** **Dipti Misra Sharma**
Language Technologies Research Centre
IIIT Hyderabad
naman.jain@students.iiit.ac.in, sambhav.jain@research.iiit.ac.in
dipti@iiit.ac.in

Abstract

In this paper, we present our efforts towards identifying probable incorrect edges and then suggesting *k*-best alternates for the same in a typed-dependency framework. Such a setup is beneficial in human aided NLP systems where the decisions are largely automated with minimal human intervention. Minimizing the human intervention calls for automatic identification of ambiguous cases. We have employed an entropy based confusion measure to capture uncertainty exerted by the parser oracle and later flag the highly uncertain predictions. To further assist human decisions, *k*-best alternatives are supplied in the order of their likelihood. Our experiments, conducted for Hindi, establish the effectiveness of the proposed approach towards increasing the label accuracy with economically viable manual intervention. This work leads to new directions for parser development and also in the human-aided NLP systems.

1 Introduction

Last decade has witnessed an increasing interest in dependency-based syntactic analysis of sentences (Tsarfaty et al., 2013). It is noticed that morphologically rich and free word order languages are better handled using the dependency based framework than the constituency based one (Mel'čuk, 1988; Bharati et al., 1995).

The fundamental notion of dependency is based on the idea that the syntactic structure of a sentence consists of binary asymmetrical relations between the words, termed as dependencies. In a typed dependency framework, the relation between a pair of words, is marked by a *dependency label*, where one of the nodes is *head* and other is *dependent* (Tesnière and Fourquet, 1959). Figure 1 shows an example sentence from Hindi, along with syntactic relations and dependency labels[1] marked along the edges.

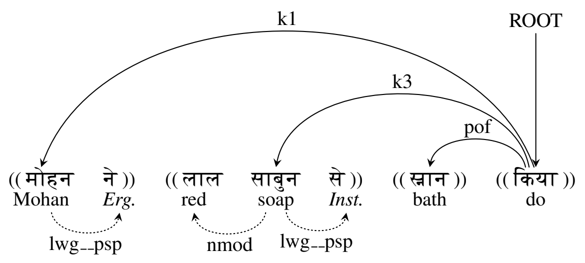

Figure 1: Dependency Tree with Syntactic Relations and Labels.

A major goal of dependency parsing research is to develop quality parsers, which can provide reliable syntactic analysis for various NLP applications such as natural language generation (Koller and Striegnitz, 2002), machine translation (Ding and Palmer, 2004), ontology construction (Snow et al., 2004), etc. Despite extensive advancements in parsing research, it is observed that parsers perform clumsily when incorporated in NLP applications (Kolachina and Kolachina, 2012). The remedies addressing the shortcomings in the past have adopted building further high quality parsers with domain adaptations (Blitzer et al., 2006; McClosky et al., 2006). However, it is practically impossible to account for all the domains and build an ideal universal parser. This has been a major reason for exploring Human Aided NLP systems which aims at providing quality output with minimal human intervention for crucial decisions.

The practical impact of parsing errors, at applica-

[1]k1: Doer, k3: Instrument, k7p: Place, pof: Part-of (complex predicate), ccof: co-ordination and sub-ordination, nmod: Noun Modifier, lwg__psp: Local-word-group post-position

tion's end, may be more severe as depicted by the accuracy of a parser. Popel (2011), in the context of machine translation, pointed out that an acutely incorrect parse can degrade the quality of output.

In this paper, we explore human assisted automated parsing, with human intervention limited to only those cases which are difficult for the statistical model (oracle) to disambiguate. Our focus has been to minimize human intervention in terms of effort and time. The scheme involves running a data driven dependency parser and later involving a human validation step for the probably incorrect decisions which are also identified and flagged by the system.

Minimizing the human intervention calls for automatic identification of ambiguous cases. We have employed an entropy based confusion measure to capture uncertainty exerted by the parser oracle and later flag the highly uncertain predictions. To further assist human decision, we also provide *k* probable alternatives in the order of their likelihood. In all, the approach comprises of the following two steps:-

- Identification of probable incorrect predictions.
- Selection of *k*-best alternates.

2 Background and Motivation

We have worked with Hindi, a relatively free-word-order and morphologically rich, Indo-Aryan language. In previous attempts to parse Hindi(Ambati et al., 2010), it has been observed that UAS[2] is greater than LS[3] by $\sim 6\%$, which is reconfirmed by our baseline parser (later described in Section 5) where UAS is 6.22% more than LS. The UAS in our baseline is well above 90% (92.44%) while the LS is still 86.21%. This drives us to focus on improving LS, to boost the overall accuracy(LA[4]) of the parser.

Dependency annotation scheme followed in Hindi Dependency Treebank (Bhatt et al., 2009) consists tag-set of $\sim$ 95 dependency labels which is comparatively larger than the tag-set for other languages[5], like Arabic($\sim$ 10), English($\sim$ 55), German($\sim$ 45) etc. This apparently is a major reason behind the observed gap between LS and UAS for Hindi parsing. One of the frequent labeling errors that the parser makes is observed to be between closely related dependency tags, for eg. $k7$ (abstract location) and $k7p$ (physical location) are often interchangeably marked (Singla et al., 2012). We have reasons to believe that such a decision is comparatively tougher for an automatic parser to disambiguate than a human validator.

In the past, annotation process has benefited from techniques like Active Learning(Osborne and Baldridge, 2004) where unannotated instances exhibiting high confusions can be prioritized for manual annotation. However, in Active Learning, the annotators or validators generally have no information about the potentially wrong sub-parts of a parse and thus full parse needs to be validated. Even if the the annotators are guided to smaller components (as in Sassano and Kurohashi (2010)), the potentially correct alternates are not endowed. In our approach the validator is informed about the edges which are likely to be incorrect and to further assist the correction k best potential label-replacements are also furnished. So, effectively just partial corrections are required and only in worst case (when a correction triggers correction for other nodes also) a full sentence needs to be analyzed. The efforts saved in our process are tough to be quantified, but the following example provides a fair idea of efficacy of our proposition. In figure 2, second parse has information of the probable incorrect label and also has 2 options to correct the incorrect label to guide a human validator.

(1) ... जेलों में समस्या हल करने ...
... prisons in problems solve do ...
... *solve problems in prisons* ...

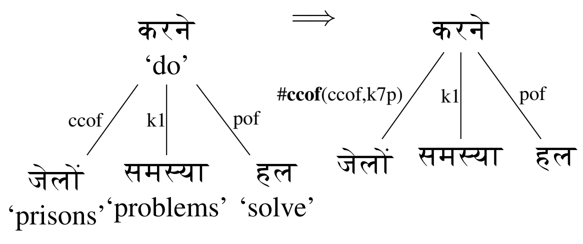

Figure 2: Example showing output from conventional parser v/s output from our approach. Arc-label with '#' represents incorrect arc label (confusion score $> \theta$) along with 2-best probable arc labels.

[2]UAS = Unlabeled Attachment Score
[3]LS = Label Accuracy Score
[4]LA = Labeled Attachment Score
[5]As observed on the CoNLL-X and CoNLL2007 data for the shared tasks on dependency parsing.

3 Methodology

Recently there has been focus on the confidence estimation of attachments and arc-labels of a dependency parse. Mejer and Crammer (2012) have worked with MSTParser (McDonald et al., 2005) to give confidence scores for attachments while Jain and Agrawal (2013) have worked with MaltParser (Nivre et al., 2007) to render the confusion scores for arc-labels. Since our focus is on arc-labels we follow the approach proposed in Jain and Agrawal (2013). They captured the confusion exerted on the parser's oracle while predicting a parser action and propagated it to the arc-label of the dependency tree. The quantification of confusion is done by calculating entropy with the class membership probabilities of the parser actions.

We obtained the confusion score for each arc-label in our data. Next, we obtained a threshold ($\theta = 0.137$) for which the maximum F_1-$score$ is observed for incorrect label identification on the development set(Figure 3). In figure 2, the edge with the label 'ccof' has been flagged (#) because the confusion score is greater than θ, which signifies that it is probably incorrect. The proposition is indeed correct as the correct label is 'k7p' instead of 'ccof'.

The additional details about the correctness of an arc-label, can duly indicate the cases where the probability of the arc-label to be incorrect is high. In our efforts to minimize the human intervention, we propose to subject the reviewer only to the cases where the confusion score is above θ. At this stage the reviewer will be required to judge if the flagged label is indeed incorrect and if it is, then choose the corresponding correct label among all the remaining labels.

To further assist human decision, we also provide *k* probable alternatives in the order of their likelihood as proposed by the oracle. The reason behind this hypothesis is that it is likely that the correct label exists among the top label candidates. This, potentially, can give quick alternates to the reviewer for choosing the correct label and thereby speedup the review process.

4 *k*-Best Dependency Labels for the Flagged Arc-Labels

The likelihood of the arc-labels is obtained and ranked using the following three strategies:-

- $Voting$: The list of predicted labels, using voting mechanism, is sorted in decreasing order of number of votes, obtained during classification. The label with maximum number of votes is emitted as the resultant dependency label in the output parse. Broadly, this can be viewed as predicting *1*-best label using *voting* strategy which can easily be extended to predict *k*-best labels.

- $Probability$: The calculation of confusion scores demand for class membership probabilities for arc-label (refer section 3). The posterior probabilities for the candidate labels can also be alternatively used to emit out the resultant dependency label. Similar to voting scheme, the labels are sorted in decreasing order of their probabilities. The sorted list of predicated labels may differ from that of *voting* mechanism, which motivate us to consider *probability* for choosing the *k*-best dependency labels.

- $Voting + Probability$: A tie can occur between two or more labels in the list of *k*-best candidate labels if their votes/posterior probabilities are same. However, the phenomenon is unlikely in case of probabilities due to the real valued nature calculated up-to 10 decimal places. On the other hand *votes* are integer-values ($\{0, ..., {}^{n}C_2\}$, where *n* is number of labels) and are much more susceptible to ties. The tie in voting can be resolved using complement information from probabilities (and vice-versa).

5 Experiments

In our experiments, we focus on correctly establishing dependency relations between the chunk[6] heads which we henceforth refer as *inter-chunk* parsing. The relations between the tokens of a chunk (*intra-chunk* dependencies) are not considered for experimentation. The decision is driven by the fact that the *intra-chunk* dependencies can easily be predicated automatically using a finite set of rules (Kosaraju et al., 2012). Moreover we also observed the high learnability of *intra-chunk* relations from a pilot experiment. We found the accuracies of *intra-chunk* dependencies to

[6]A chunk is a set of adjacent words which are in dependency relation with each other, and are connected to the rest of the words by a single incoming arc.

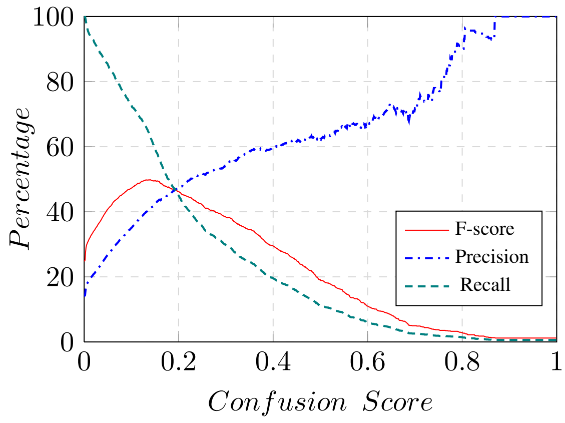

Figure 3: *Precision*, *Recall* and F_1-*score* for various values of confusion score on 'Hindi' development set.

be more than 99.00% for both LA and UAS.

Each experiment assumes the availability of a human expert for validation of the machine parsed data, who, when queried for a potential incorrect edge label, responds with the correct edge label. The experiments aim to measure the assistance provided to human expert by our approach. We varied the list of k-best labels from k=1 to k=5.

We setup a baseline parser on the lines of Singla et al. (2012) with minor modifications in the parser *feature model*. We employ MaltParser version-1.7 and Nivre's Arc Eager algorithm for all our experiments reported in this work. All the results reported for overall parsing accuracy are evaluated using *eval07.pl*[7]. We use MTPIL (Sharma et al., 2012) dependency parsing shared task data. Among the features available in the FEATS column of the CoNLL format data, we only consider *Tense, Aspect, Modality (tam)*, *post-positions(vib)* and *chunkId* while training the baseline parser. Other columns like POS, LEMMA, etc. are used as such.

In case of typed-dependency parsing, the accuracy can be LA, UAS or LS. However, in our case, we are focusing on the correct prediction of arc-labels, the results are on LS. In terms of strategies mentioned in Section 4, baseline system is generated using *Voting* strategy with $k = 1$. The LS is 86.21% as shown in Table 1.

[7]http://nextens.uvt.nl/depparse-wiki/SoftwarePage/#eval07.pl

6 Evaluation and Discussion

The evaluation of an interactive parse correction is a complicated task due to intricate cognitive, physical and conditional factors associated with a human annotator. Since a human may not match the consistency of a machine, we have to resort to few compelling assumptions which would give an idea of the approximate benefit from our proposed approach. We have assumed a perfect human oracle who always identifies incorrect label and picks the correct label from the available k-best list, if correct label is present in the list. The simulation of the perfect human oracle is done using the gold annotation. It is also assumed that the decision of the correct label can be taken with the information of local context and the whole sentence is not reviewed(which is not always true in case of a human annotator). This gives the upper bound of the accuracies that can be reached with our approach. The validation of the results obtained by automatic evaluation is done by performing a separate human evaluation for 100 nodes with the highest confusion score.

In our dataset, we found $\sim$23% (4,902 edges) of total (21,165) edges having confusion score above θ and thus marked as potentially incorrect arc-labels. Table 1 exhibits LS improved by perfect human oracle, for k-best experiments where k=1 to 5 on $\sim$23% potentially incorrect identified arc-labels.

k	Voting(%)	Probability(%)	Voting+Probability(%)
1	86.21	**86.35**	86.28
2	90.86	**90.96**	90.91
3	92.13	**92.24**	92.18
4	92.72	**92.86**	92.74
5	92.97	**93.16**	93.04

Table 1: k-Best improved LS on inspecting $\sim$23% ($> \theta$) edges.

Table 1 also depicts that as the value of k increases, the label accuracy also increases. The best results are obtained for *Probability* scheme. There is a substantial increment in LS moving from 1-best to 2-best in all the schemes. The amount of gain, however, decreases with increase in k.

Ideally to achieve maximum possible LS, all the edges should be reviewed. Table 2 confirms that if all the edges are reviewed, an LS of 93.18% to 96.57% is achievable for k, ranging over 2 to 5. But practically this would be too costly in terms of time and effort. In order to economize, we wish to only review the cases

k	**Voting(%)**	**Probability(%)**	**Voting+Probability(%)**
1	86.21	**86.35**	86.28
2	93.19	93.18	**93.26**
3	95.10	95.11	**95.14**
4	95.94	**96.06**	95.97
5	96.41	**96.57**	96.49

Table 2: *k*-Best improved LS on inspecting 100% edges.

which are probable enough to be incorrect. Confusion scores give a prioritized list of edges, which dictates the cases that should be dispatched first for review. To relate the review cost against LS gain we present a metric AGI$_x$ defined as:-

AGI$_x$:*"Accuracy Gain on Inspecting top $x\%$ edges"* corresponds to the ratio of accuracy gain from baseline by inspecting top $x\%$ of total edges, when sorted in decreasing order of their *confusion score*. The metric takes into account the human effort that goes into validation or revision, and thus gives a better overview of ROI(Return on Investment).

$$AGI_x = \frac{\text{Accuracy after validating top } x\% \text{ edges} - \text{Baseline accuracy}}{x}$$

From Table 1 and Table 2 we observe for $k = 2$ and *probability* scheme that the improved LSs are 90.96% and 93.18% on inspecting 23% and 100% edges respectively. Although the latter is greater than former by $\sim$ 2% but this additional increment requires an extra inspection of additional $\sim$77% edges, which is economically inviable. The fact is better captured in Table 3, where AGI$_{23}$ subdues AGI$_{100}$ for different values of *k* using different 'schemes'.

Further to incorporate the fact that *'larger the candidate list more will be the human efforts required to pick the correct label'*, we also present the results of **AGI$_x$/k**, which can govern the choice of *k*, best suited in practice. While taking this into account, we assume that the human efforts are inversely proportional to *k*. Results for **AGI$_{23}$/k** on improved LS, over all the experiments are reported in Table 4.

As shown in Table 3, AGI$_{23}$ increases with increase in the value of *k*, but it is practically inefficient to keep large value of *k*. Optimum choice of *k* is observed to be 2 from the metric AGI$_x$/k, as shown in Table 4, where the maximum value for **AGI$_{23}$/k** is $\sim$ 0.10 for all the 'schemes', which corresponds $k = 2$.

	Voting		**Probability**		**Voting+Probability**	
k	**AGI$_{23}$**	**AGI$_{100}$**	**AGI$_{23}$**	**AGI$_{100}$**	**AGI$_{23}$**	**AGI$_{100}$**
1	0.0000	0.0000	**0.0060**	0.0014	**0.0030**	0.0007
2	**0.2008**	0.0698	**0.2051**	0.0697	**0.2029**	0.0705
3	**0.2556**	0.0889	**0.2604**	0.0890	**0.2578**	0.0893
4	**0.2811**	0.0973	**0.2871**	0.0985	**0.2820**	0.0976
5	**0.2919**	0.1020	**0.3001**	0.1036	**0.2949**	0.1028

Table 3: ***AGI$_{23}$*** and ***AGI$_{100}$*** for *k*=1 to 5

k	**AGI$_{23}$/k** (Voting)	**AGI$_{23}$/k** (Probability)	**AGI$_{23}$/k** (Voting+Probability)
1	0.0000	0.0060	0.0030
2	**0.1004**	**0.1025**	**0.1015**
3	0.0852	0.0868	0.0859
4	0.0703	0.0718	0.0705
5	0.0584	0.0600	0.0590

Table 4: ***AGI$_{23}$/k*** for *k*=1 to 5

From the above analysis, we can establish that with 2 probable alternatives, a perfect human oracle can increase the LS by 4.61%, inspecting top $\sim$ 23% of total edges. The corresponding LA increase is 4.14%(earlier 83.39% to now 87.53%).

The validation of the observation is done by a human expert who confirmed of the assistance from the above methodology over the default procedure. He was given with 2-best alternatives for the top 100 edges that are obtained using *probability* scheme. The LS gain on his evaluation is approximately 10% which matches the expected gain.

7 Conclusion

In this paper we explored the possibility of human intervention to achieve higher accuracies in parsing. A major hurdle in the process is to effectively utilize the valuable human resources. We employed an entropy based confusion measure to capture uncertainty exerted by the parser oracle and later flag the highly uncertain labels. We further asserted that with 2 probable alternatives, a human expert can increase the label accuracy by 4.61%, inspecting $\sim$ 23% of total edges. In future we would also like to study the effectiveness of our approach on attachment validation in parsing.

References

A. Bharati, V. Chaitanya, R. Sangal, and KV Ramakrishnamacharyulu. 1995. *Natural Language Processing: A Paninian Perspective*. Prentice-Hall of India.

Alexander Koller and Kristina Striegnitz. 2002. Generation as dependency parsing. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 17–24.

Avihai Mejer and Koby Crammer. 2012. Are you sure?: confidence in prediction of dependency tree edges. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 573–576. Association for Computational Linguistics.

Bharat Ram Ambati, Samar Husain, Sambhav Jain, Dipti Misra Sharma, and Rajeev Sangal. 2010. Two methods to incorporate local morphosyntactic features in Hindi dependency parsing. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 22–30. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. 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*, pages 152–159. Association for Computational Linguistics.

Dipti Misra Sharma, Prashanth Mannem, Joseph vanGenabith, Sobha Lalitha Devi, Radhika Mamidi, and Ranjani Parthasarathi, editors. 2012. *Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages*. The COLING 2012 Organizing Committee, Mumbai, India, December.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kubler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128. Association for Computational Linguistics.

Karan Singla, Aniruddha Tammewar, Naman Jain, and Sambhav Jain. 2012. Two-stage approach for hindi dependency parsing using maltparser. *Training*, 12041(268,093):22–27.

Lucien Tesnière and Jean Fourquet. 1959. *Eléments de syntaxe structurale*, volume 1965. Klincksieck Paris.

Manabu Sassano and Sadao Kurohashi. 2010. Using smaller constituents rather than sentences in active learning for japanese dependency parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 356–365.

Martin Popel, David Mareček, Nathan Green, and Zdeněk Žabokrtskỳ. 2011. Influence of parser choice on dependency-based MT. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 433–439. Association for Computational Linguistics.

Mel'čuk. 1988. *Dependency syntax: theory and practice*. State University Press of New York.

Miles Osborne and Jason Baldridge. 2004. Ensemble-based active learning for parse selection. In *HLT-NAACL*, pages 89–96.

Prudhvi Kosaraju, Samar Husain, Bharat Ram Ambati, Dipti Misra Sharma, and Rajeev Sangal. 2012. Intra-chunk dependency annotation: expanding Hindi inter-chunk annotated treebank. In *Proceedings of the Sixth Linguistic Annotation Workshop*, pages 49–56. Association for Computational Linguistics.

R. Bhatt, B. Narasimhan, M. Palmer, O. Rambow, D.M. Sharma, and F. Xia. 2009. A multi-representational and multi-layered treebank for hindi/urdu. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 186–189. Association for Computational Linguistics.

Reut Tsarfaty, Djamé Seddah, Sandra Kübler, and Joakim Nivre. 2013. Parsing Morphologically Rich Languages: Introduction to the Special Issue. *Computational Linguistics*, 39(1):15–22.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems 17*.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 523–530. Association for Computational Linguistics.

Sambhav Jain and Bhasha Agrawal. 2013. A dynamic confusion score for dependency arc labels. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1237–1242, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Sudheer Kolachina and Prasanth Kolachina. 2012. Parsing any domain english text to conll dependencies. In *LREC*, pages 3873–3880.

Yuan Ding and Martha Palmer. 2004. Synchronous dependency insertion grammars: A grammar formalism for syntax based statistical mt. In *Workshop on Recent Advances in Dependency Grammars (COLING)*, pages 90–97.