Evaluation of Two Bengali Dependency Parsers

Arjun Das Arabinda Shee Utpal Garain
INDIAN STATISTICAL INSTITUTE, 203, B. T. Road, Kolkata 700108, India.
{arjundas|utpal}@isical.ac.in

ABSTRACT

In this paper we have addressed two dependency parsers for a free-word order Indian language, namely Bengali. One of the parsers is a grammar-driven one whereas the second parser is a data-driven one. The grammar-driven parser is an extension of a previously developed parser whereas the data driven parser is the MaltParser customized for Bengali. Both the parsers are evaluated on two datasets: ICON NLP Tool Contest data and Dataset-II (developed by us). The evaluation shows that the grammar-based parser outperforms the MaltParser on ICON data based on which the demand frames of the Bengali verbs were developed but its performance degrades while dealing with completely unknown data, i.e. dataset-II. However, MaltParser performs better on dataset-II and the whole data. Evaluation and error analysis further reveals that the parsers show some complimentary capabilities, which indicates a future scope for their integration to improve the overall parsing efficiency.

KEYWORDS: Dependency parser, MaltParser, Grammar-driven parser, ICON data, Treebank, Paninian Grammatical model, Demand Groups, Free-word order language, Bengali.

Proceedings of the Workshop on Machine Translation and Parsing in Indian Languages (MTPIL-2012), pages 133–142,
COLING 2012, Mumbai, December 2012.

1 Introduction

Most of the Indian languages including Bengali have relatively free-word order [Bharati et al., 1995]. This characteristic makes the dependency parsing of Bengali a challenging task. There are two approaches used for dependency parsing, data driven dependency parsing and grammar driven dependency parsing approach [Nivre, 2006]. Data driven dependency parsing requires large amount of manually annotated parsed data. On the other hand grammar driven dependency parsing requires a set of linguistics rules. Systematic evaluation of these parsing approaches was not done until ICON, in 2009, conducted an NLP tool contest on development of dependency parsers for three Indic languages namely, Hind, Bengali, and Telugu [ICON 2009]. Training, development and test data was provided. Same shared task was repeated in ICON 2010. Both grammar driven and data driven approaches were reported.

This paper presents our continued effort that started with participating in ICON 2009. A grammar driven parser [De et al., 2009] for Bengali was developed and achieved an unlabeled attachment score close to 90%. The linguistics rules were extracted from the ICON 2009 training data set and tested on ICON 2009 test set. This parser was not tested on any unknown dataset which was not used in extracting linguistics rules. This paper aims to fill up that gap by generating a new dataset consisting of tree banks for about 1500 sentences. Evaluation on this new dataset is done. Moreover, the previous parser was improved by adding and optimizing certain rules. Next, an off the shelf parser namely, MaltParser [Nivre et al., 2006a] is used to parse Bengali and a comparison between grammar driven and data driven approaches is brought out.

2 The Grammar-driven Parser

The Paninian Grammatical model, which is very effective for free-word order languages (e.g. Indian Languages) has been used for development of a grammar-driven parser. The approach is to simplify complex and compound sentential structures first, then to parse the simple structures so obtained by satisfying the Karaka demands of the Demand Groups (Verb Groups) and to rejoin such parsed structures with appropriate links and Karaka labels. The parsing algorithm is given below.

```
Input: A sentence with all morphological and chunking information.
Output: A dependency tree having the chunked phrases as nodes.

Step-1: If the sentence is compound then divide the sentence to get two
       or more simple or complex sentences. Pass each of them to Step 2
       one by one.
       Otherwise pass the sentence to Step 2.
Step-2: If the sentence is complex then divide the sentence to get two
       or more simple sentences. Pass each of them to Step 3 one by one.
       Otherwise pass the sentence to Step 3.
Step-3: Parse the simple sentence
Step-4: Rejoin the parsed sentences divided in step 2 with proper links
       and labels.
Step-5: Rejoin the parsed sentences divided in step 1 with proper link
       and label.
Step-6: Return the parsed sentence.
```

The sentences which have coordinate conjuncts have been treated as compound sentences and handled in Step-1 of the algorithm. Consider the sentence below.

S1. (রাম ভাত খায়) এবং (শ্যাম রুটি খায়)
(Ram bhat khay) ebam (Shyam ruti khay)
(Ram rice eats) and (Shyam bread eats.)

In the above sentence, two simple sentences shown within braces are joined with sentence label conjunct ebam (and) to form a compound sentence. Our approach is to identify these sentence label conjuncts and divide the sentence to make the parsing task easier. After parsing the two simple sentences the roots of the two sentences are linked with the conjunct with 'ccof' relation. The sentences having relative clauses are considered as complex sentences and handled in Step 2 of the Algorithm. Consider the sentence below.

S2. (যে ছেলেটি সেখানে বসে আছে) (সে আমার ভাই হয়)
(je chheleti sekhane base achhe) (se amar bhai hay)
(Who boy the there sitting is) (he my brother is)

The first part of the sentence is a relative clause which modifies se (he), je (who) and se (he) are grammatical markers of relative clause and main clause, respectively. With the help of the clause markers, a complex sentence is divided into multiple simple sentences which are then parsed in Step-3 and rejoined in Step-4.

Simple sentences have been parsed with demand satisfaction approach. A Demand Frame or Karaka Frame of a verb is a tabular listing of the demands it makes i.e. the list of all possible Karakas it can take to form a meaningful sentence [Begum, 2008]. A mapping is also specified in the list between Karaka relations and Vibhaktis (post-positions, suffix). The mapping depends on the verbal semantics and the tense, aspect and modality (TAM) label. The basic frame or default frame of a verb is prepared for present indefinite form of the verb [Bhattacharya 1993]. Other TAMs may have their own transformation rules depending on which the basic frame is changed. The Demand Frame of a verb also specifies what Karakas are mandatory or optional for the verb and what Vibhaktis (post-positions) they take. Thus, for a given verb with some TAM label, the appropriate Karaka frame can be obtained using the basic frame and the corresponding transformation rules. The basic frame which is initially prepared for the present indefinite form of a verb may change with the change of the form of the verb i.e. TAM labels. We have prepared an exhaustive TAM list in Bangla and transformation rules, if exists, have been framed for each of them.

For a given sentence after the word groups have been formed, the verb groups are identified [Biswas et al., 2010; De et al. 2011]. Then each of the source groups are tested against the Karaka restrictions in each Karaka frame (transformed according to TAM rules). When testing a source group against the Karaka restrictions of a demand group, vibhaki information is checked and if found satisfactory the source group becomes a candidate for the Karaka of the demand group. This can be shown in the form of a Constraint Graph (CG) [Bharati et. al. 2008 and 2009]. Nodes of the graph are the word groups and there is an arc from a verb group to a source group labeled by a Karaka, if the source group satisfies the Karaka restrictions in the Karaka chart. A restricted CG can be obtained by following certain rules involving gnp (gender, number, person) agreement, matching of lexical types, etc. The detailed of the parsing approach can be found in [De et al., 2009; Garain et al., 2012]

3 MaltParser for Bengali

In this experiment we have customized freely available MaltParser [Nivre et al., 2006a] which follows a data-driven approach. During MaltParser optimization we follow same approach described by Nivre, (2009). MaltParser comes with a number of built in transition systems. So we evaluate different transition systems and found that stackswap transition system gave the highest accuracy for Bengali. In order to tune feature model we first added all possible features. Then we discarded those features for which the parsing accuracy increases. Finally, we end up with addition of following feature numbers:

- Features 3 and 10, the coarse-grained part of speech of top and next.
- Features 22 and 25, the part of speech of leftmost and rightmost dependencies of top.
- Features 24, the form of rightmost dependencies of top.
- The conjoined features (1&4, 4&11, 8&11) i.e. part of speech and form of stack top, part of speech of top and next, form of next and part of speech of next was also added.

We used LIBSVM package [Chang and Lin, 2001] for classification task.

3.1 Training Data

To train MaltParser we need parsed Treebank in CoNLL format which is developed during this research. For this purpose we converted ICON SSF data into CoNLL format. Additionally, we have taken about 1555 sentences from a list of 9 stories and parse them automatically using the grammar-based Parser. Then the parsed Treebank (i.e. SSF format) was corrected thoroughly by a linguist. We call this dataset as DS-II. The combination of ICON and DS-II resulted in a Treebank consisting of about 2685 parsed sentences (29137 tokens) in SSF and CoNLL format. The CoNLL fields which we used for feature models are: ID, FORM, POSTAG, CPOSTAG, HEAD and DEPREL.

4 Evaluation

We evaluated the above parsers following the methods described in [Rimell et al. 2009] and [Nivre et al. 2010]. The first evaluation considers ICON (2009) data sets (i.e. both training and development data of 1130 Sentences). The grammar-driven parser uses this dataset to extract linguistic rules. The Malt Parser is trained with this data. Evaluation of the parsers is done using ICON 2009 test data (consisting of 150 Sentences). As expected, both the Malt Parser and the grammar driven parser achieved scores close to [Nivre, 2009; De et al., 2009], respectively. Table 1 shows the evaluation results of both parsers on ICON 2009 datasets.

Table 1. Performance on ICON 2009 Data

	MaltParser			Grammar driven Parser		
	LAS	**UAS**	**LA**	**LAS**	**UAS**	**LA**
Bengali-Coarse	67.75	82.08	63.83	84.29	90.32	85.95
Bengali-Fine	57.67	82.64	60.13	79.81	90.32	81.27

Our second experiment considers the larger dataset adding DS-II (prepared by us) to ICON data. A list of 2600 sentences (i.e. 30359 tokens) is divided into 5 sets to facilitate a 5-fold cross validation. The ratio of training and test data is 4:1. Each set is used once as a test data. Malt Parser is retrained on this data but the grammar driven parser was not retrained. Table 2 shows the parsers accuracies (on coarse tag set) after the execution of the 5-fold cross validation. Baseline accuracy represents the performance of Malt Parser with default properties where final accuracy is achieved after optimization of Malt Parser.

Table 2 Parsing accuracy on Larger Dataset

	MaltParser			Grammar driven Parser		
	LAS	**UAS**	**LA**	**LAS**	**UAS**	**LA**
Baseline	52.50	71.01	56.11	51.39	63.23	54.32
Final(Optimized)	56.61	76.31	59.91			

If we compare the results in Table 1 and 2, the grammar driven parser performs well on ICON data but its performance degrades on the whole dataset. The major reason is that the linguistics rules were extracted from ICON data only. The demand frames of the verbs were constructed based on the examples found in ICON data. The degradation in performance shows that some verbs are missing in the verb list. Demand frames of some verbs are also incomplete. Another reason is the use of morphological information. ICON data is annotated with morphological information which was used by the grammar driven parser. But as the DS-II does not contain morphological information, parser itself attempts to extract this information. In some cases this additional processing adds errors.

Training of MaltParser did not use the morphological information as available with ICON data. It makes use of POS tag, chunking information and dependency relations. On the bigger dataset its performance degrades but the amount of degradation is less compared to the degradation shown by the grammar-driven parser.

4.1 Error Analysis

A primary goal of this experiment is to point out the errors made by both data driven and the grammar driven dependency parsing model. Following [McDonald et al. 2011] we have performed a number of experiments to find out the reasons of most possible errors with respect to sentence length factor and linguistics properties of sentences.

4.1.1 Length Factor

Length factor is a well known factor. It is noticed that dependency parsing system tends to have lower accuracy for longer sentences. The main reason is that a longer sentence is a combination of two or more simple shorter sentences. Figure 1 shows the

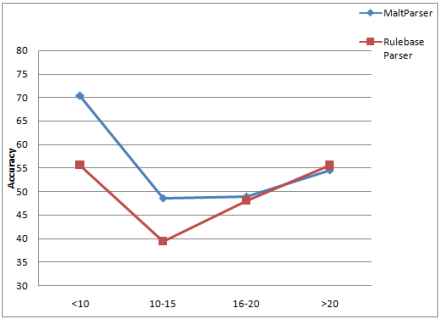

Figure 1 Parser Accuracy (LAS) and Sentence Length

accuracy i.e. the labelled attachment score (LAS) of both parsing model with respect to different sentence length, i.e. number of tokens. From the figure it is clear Malt Parser tends to perform better than the grammar driven Parser on shorter sentences, due to greedy inference algorithm which make fewer parsing decision. Otherwise system performance of both parsers is indistinguishable. Surprisingly accuracy of longer sentences (Sentence length >16) on both parsers seems to be improved. This is due to the low complexity properties of the sentences.

4.1.2 Dependency Relation Wise Evaluation

Table 3 presents accuracy of the parsers per dependency relation. Meaning of the dependency relation tags can be found in [ICON 2009]. The table lists down accuracies for 24 dependencies. It is observed that Malt Parser shows better recall whereas the grammar driven parser shows better precision. Malt Parser shows better recall for 13 out of 24 dependencies whereas the grammar driven parser shows better recall for only 8 dependency relations. However, if precision is concerned the grammar driven parser shows better performance for 17 dependency relations. The Malt parser shows better precision for only 5 dependency relations.

Table 3. Dependency Relation-wise Performance Evaluation

Deprel	Gold	MaltParser				Grammar-driven Parser			
		correct	system	recall	Precision	correct	system	recall	precision
ROOT	991	900	1000	90.82	90.00	826	1988	83.35	41.55
Ccof	606	495	654	81.68	75.69	445	607	73.43	73.31
k*u	8	2	4	25.00	50.00	0	0	0.00	NaN
k1	848	559	1106	65.92	50.54	497	835	58.61	59.52
k1s	125	50	92	40.00	54.35	65	188	52.00	34.57
k2	808	476	1051	58.91	45.29	279	615	34.53	45.37
k2s	30	2	25	6.67	8.00	4	4	13.33	100.00
k3	15	0	6	0.00	0.00	4	7	26.67	57.14
k5	33	11	11	33.33	100.00	11	11	33.33	100.00
k7	102	11	23	10.78	47.83	7	10	6.86	70.00
k7p	204	53	182	25.98	29.12	32	61	15.69	52.46
k7t	215	99	166	46.05	59.64	60	82	27.91	73.17
Nmod	87	6	17	6.90	35.29	6	6	6.90	100.00
nmod__relc	34	0	1	0.00	0.00	8	8	23.53	100.00
nmod_relc	1	1	7	100.00	14.29	0	10	0.00	0.00
Pof	203	97	197	47.78	49.24	73	89	35.96	82.02
r6	496	268	366	54.03	73.22	402	523	81.05	76.86
Rad	72	17	29	23.61	58.62	3	3	4.17	100.00
ras-k2	12	0	0	0.00	NaN	2	2	16.67	100.00
Rd	2	0	4	0.00	0.00	0	4	0.00	0.00
Rh	20	0	3	0.00	0.00	4	4	20.00	100.00
Rs	10	0	0	0.00	NaN	2	2	20.00	100.00
Rt	16	2	7	12.50	28.57	1	1	6.25	100.00
Vmod	552	385	603	69.75	63.85	278	362	50.36	76.80

4.1.3 Sentence wise comparison

We have tried to mark the sentences where the parsers show complementary results. Several cases of this type have been identified. The following is an example where the Malt parser produces the correct parse but the grammar-driven parser fails.

S3. গণপতিবাবু এবার কপালে হাত ঠেকালেন ।
Ganapatibabu then on forehead hand touched.

Result from Malt Parser:

1	গণপতিবাবু	_	NP	NNP	_	5	k1	_	_
2	এবার	_	NP2	NN	_	5	k7t	_	_
3	কপালে	_	NP3	NN	_	5	k7p	_	_
4	হাত	_	NP4	NN	_	5	k2	_	_
5	ঠেকালেন	_	VGF	VM	_	0	ROOT	_	_
6	।	_	VGF	SYM	_	0	ROOT	_	_

Result from the Grammar driven Parser:

1	গণপতিবাবু	_	NP	NNP	_	5	k1
2	এবার	_	NP2	NN	_	5	k7t
3	কপালে	_	NP3	NN	_	0	ROOT
4	হাত	_	NP4	NN	_	0	ROOT
5	ঠেকালেন	_	VGF	VM	_	0	ROOT
6	।	_	VGF	SYM	_	0	ROOT

The grammar driven parser fails to identify the root properly.

Examples are there where the grammar driven parser produces correct result but the Malt parser fails. Here is one example that has a compound verb.

S4. সে তখন ড্রেসিং গাউন পরে তার বাসগৃহের লনে পায়চারী করছে ।

(he)(then)(dressing)(gown)(wearing)(his)(residence)(at lawn)(walking was).

Result (in correct) from Malt Parser:

1	সে	_	NP	PRP	_	10	k1
2	তখন	_	NP2	PRP	_	10	k7t
3	ড্রেসিং	_	NP3	XC	_	5	k2
4	গাউন	_	NP3	NN	_	5	k2
5	পরে	_	VGNF	VM	_	10	vmod
6	তার	_	NP4	PRP	_	7	r6
7	বাসগৃহের	_	NP5	NN	_	8	r6
8	লনে	_	NP6	NN	_	10	k1
9	পায়চারী	_	NP7	NN	_	10	pof
10	করছে	_	VGF	VM	_	0	ROOT
11	।	_	VGF	SYM	_	0	ROOT

Result (correct) from the Grammar-driven Parser:

1	সে	_	NP	PRP	_	10	k1
2	তখন	_	NP2	PRP	_	5	k7t
3	ড্রেসিং	_	NP3	XC	_	5	k2
4	গাউন	_	NP3	NN	_	5	k2
5	পরে	_	VGNF	VM	_	10	vmod
6	তার	_	NP4	PRP	_	7	r6
7	বাসগৃহের	_	NP5	NN	_	8	r6
8	লনে	_	NP6	NN	_	0	k7p
9	পায়চারী	_	NP7	NN	_	10	pof
10	করছে	_	VGF	VM	_	0	ROOT
11	।	_	VGF	SYM	_	0	ROOT

Acknowledgement

The authors sincerely thank the Society for Natural Language Technology Research (SNLTR) for their partial financial support for carrying out the research embodied in this paper.

Conclusions

This paper presents an evaluation of two parsers for Bengali. One parser is grammar driven and the second one is data driven. Evaluation shows that the grammar driven parser can be improved further if the demand frames for Bengali verbs can be made complete. Malt parser did not use any morphological information and use of this information as features could improve parsing efficiency. The parsers show some complimentary performance in terms of recall and precision and also in parsing different sentences. Integration of these two parsers can be done in future in order to improve the overall parsing efficiency. The present research has extended ICON data by adding about 1500 annotated sentences and this can be considered as a good NLP resource for future use. The evaluation results reported here will also serve as a useful finding for conducting future research in this area.

References

Begum R, Husain S., Sharma D.M., and Bai L., (2008). Developing Verb Frames for Hindi, In the Proc LREC 2008.

Bharati, A., Sangal, R. (1995). Parsing Free Word Order Languages in the Paninian Framework. Proc. of ACL.

Bharati A., Husain S., Sharma D.M., and Sangal R., (2008). A Two-Stage Constraint Based Dependency Parser for Free Word Order Languages. In Proc. of the COLIPS International Conference on Asian Language Processing 2008 (IALP), Thailand, 2008.

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

Biswas, S., Dhar, A., De, S., and Garain, U. (2010). Performance Evaluation of Text Chunking. In Proc. of 8th Int. Conf. on Natural Language Processing (ICON), Kharagpur, India.

Chih-Chung Chang and Chih-Jen Lin, (2001). LIBSVM: A Library for Support Vector Machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

De, S., Dhar, A., and Garain, U. (2009). Structure Simplification and Demand Satisfaction Approach to Dependency Parsing for Bangla. In Proc. of 6th Int. Conf. on Natural Language Processing (ICON) tool contest: Indian Language Dependency Parsing, 25-31, India.

De, S. Dhar, A. Biswas, S. and Garain, U. (2011). On Development and Evaluation of a Chunker in Bangla. In IEEE Proc. of 2nd Int. Conf. on Emerging Applications of Information Technology (EAIT), 321-324, Kolkata, India.

Garain, U. and De, S. (2012). Dependency Parsing in Bangla. In Technical Challenges and Design Issues in Bangla Language Processing, Eds: M. A. Karim, M. Kaykobad, and M. Murshed, Publisher: IGI Global, USA (in Press).

ICON (2009). NLP Tool Contest: Parsing, In 7th International Conference on Natural Language Processing (ICON), Hyderabad, India.

McDonald, R. and Nivre, J. (2011) Analyzing and Integrating Dependency Parsers. Computational Linguistics 37(1), 197-230.

Michael A. Covington, (2001). A fundamental algorithm for dependency parsing. In Proceedings of the 39th Annual ACM Southeast Conference, pages 95-102.

Nivre, J. (2006). Inductive Dependency Parsing, Springer.

Nivre, J., J. Hall, and J. Nilssion. (2006a). MaltParser: A data-driven parser-generator for dependency parsing. In Proceedings of LREC, 2216-2219.

Nivre, J. (2009). Parsing Indian Languages with MaltParser. In Proceedings of the ICON09 NLP Tools Contest: Indian Language Dependency Parsing, 12-18.

Nivre, J., Rimell, L., McDonald, R., and Gomez-Rodriguez, C. (2010). Evaluation of dependency parsers on unbounded dependencies. In Proceedings of the 23rd International Conference on Computational Linguistics (COLING '10). Association for Computational Linguistics, Stroudsburg, PA, USA, 833-841.

Rimell, L., S. Clark, and M. Steedman. (2009). Unbounded dependency recovery for parser evaluation. In Proceedings EMNLP, 813-821.