Grammar Driven Rules for Hybrid Bengali Dependency Parsing

Sanjay Chatterji
Computer Science and Engineering
IIT Kharagpur
Kharagpur, W.B., India
sanjaychatter@gmail.com

Praveen Sonare
Computer Science and Engineering
IIT Kharagpur
Kharagpur, W.B., India
praveensonare@gmail.com

Sudeshna Sarkar
Computer Science and Engineering
IIT Kharagpur
Kharagpur, W.B., India
sudeshna@gmail.com

Devshri Roy
Computer and Informatics Centre
IIT Kharagpur
Kharagpur, W.B., India
droy@cse.iitkgp.ernet.in

Abstract

This paper describes a hybrid approach for parsing Bengali sentences based on the dependency tagset and Treebank released in ICON 2009 tool contest. A data driven dependency parser is considered as a baseline system. Some handcrafted rules are identified based on the error patterns on the output of the baseline system.

1 Introduction

Parsing is a method of analyzing the grammatical structure of a sentence. Dependency parsing indicates the dependency relations between the words in a parse tree. Data driven and grammar driven approaches have been used for parsing a sentence. Data driven parsers need large amount of manually annotated parsed data which is called a Treebank. The availability of such data for Bengali is limited. A data driven parser which was trained on the Bengali Treebank data released by ICON tool contest, 2009 was found to have limited accuracy. On the other hand, most of the modern grammar-driven dependency parsers (Karlsson et al.; 1995, Bharati et al.; (1993, 2002, 2008)) parse by eliminating the parses which do not satisfy the given set of constraints. They require rules to be developed for each layer. In the work of Akshar Bharati et al. (PACLIC, 2009; IWPT09, 2009), the grammar driven approach is complemented by a controlled statistical strategy to achieve high performance and robustness.

Developing a grammar driven parser requires a deep knowledge of the dependency relations of the phrases. The creation of rules to disambiguate the relations is very challenging task. We use a hybrid approach for shallow level dependency parsing of Bengali texts where the output of the data driven parser is postprocessed by applying certain rules to improve the parser accuracy.

Bengali is the seventh widely spoken language in the world (Ethnologue, 2009). The work done on parsing Bengali text is quite limited. In this paper, we discuss a hybrid system developed for shallow level dependency parsing of Bengali sentences. The system uses an openly available data driven dependency parser MaltParser version 1.2 (Nivre et. al., 2006) as a baseline system. The Bengali Treebank released by ICON tool contest, 2009 is used for training. We have suggested few rules based on the error patterns of the MaltParser output. The output of the baseline system is postprocessed based on these rules to improve the accuracy of the system. These rules are not complete in handling all the error patterns. More study is required to prepare a comprehensive rule set to further improve the results.

2 System Architecture

Figure 1 shows the system architecture of our system. The freely available MaltParser version 1.2 is used as a baseline system. The MaltParser

Proceedings of ICON-2009: 7th International Conference on Natural Language Processing,
Macmillan Publishers, India. Also accessible from http://ltrc.iiit.ac.in/proceedings/ICON-2009

is trained with the training data released by the ICON tool contest 2009. The trained parser is tested with the development data given by ICON tool contest 2009. The data is tagged based on the dependency relations given in AnnCorra(Sharma et. al.) tagset. We have done the postprocessing part in 4 stages. The post-processing stages are also carried out on the output of the MaltParser on the development data. These four stages are explained in section 4. We have evaluated the output in each stage using the official CoNLL-07 shared task evaluation script eval07.pl.

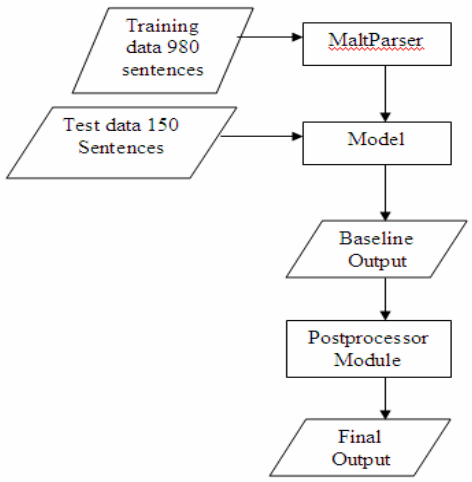

Figure 1: System Architecture

3 Confusion Matrix

As discussed in Section 2, the MaltParser is trained and tested with the given data set. The base line score obtained is 54.62% for labeled attachment, 79.28% for unlabeled attachment. We examined the confusion matrix and identified the tags for which the most errors have been made. Then we worked out postprocessing rules to correct some of the errors. The original confusion matrix is a 28×28 matrix. A portion of the confusion matrix is given in Table 1. The tag nomenclature meanings are as follows: k1 – Karta (doer/agent/subject), k2 – Karma (object), k7p – Deshadhikarana (location in space), r6 – Shashthi (possessive), pof – Part of relation, k7t – kaalaadhikarana (location in time), nmod_relc– Noun modifier of the type relative clause, rt– Taadarthya(Purpose), vmod–Verb Modifier, k7– Vishayaadhikarana(Location elsewhere) and k5– Apaadaana(Source).

	k1 (Identified)	k2 (Identified)	k7p (Identified)	r6 (Identified)	pof (Identified)
k1 (True)	92	23	4	9	14
k2 (True)	10	47	0	0	14
k7p (True)	12	5	11	2	7
r6 (True)	14	5	2	29	0
pof (True)	1	4	0	0	18

Table1: A portion of Confusion Matrix

In the confusion matrix, the rows represent the true tags and the columns represent the tags identified by the parser. For example, the true tag k1 appears as 92 times k1, 23 times k2, 4 times k7p, 9 times r6 and 14 times pof in the MaltParser output data. The diagonal elements represent correctly identified tags while the non-diagonal elements represent wrongly identified tags. Note the pof(Identified) column of the matrix. Among all the *pof* MaltParser has identified 18 *pof* correctly. 14 *k1* and 14 *k2* are wrongly tagged as pof. 7 *k7p* are also wrongly tagged as *pof*.

In order to improve the accuracy we try to minimize the non-diagonal figures in the matrix. We observed that in many cases, tags “k1” and “k2” and “k7p” were identified as “pof” and in some cases, they were identified as “r6”. We also observed that in many cases, the parser did not identify the root of the sentence correctly.

To minimize the non-diagonal figures of the matrix, the parsed output data is observed carefully to discover different rules and the MaltParser output data is postprocessed.

4 Postprocessing

The output of the MaltParser is postprocessed in four steps to improve the quality of the parsed output. In the first step, we tried to identify the phrases from the sentences that are the root of the parse tree. We have discussed a TAM marker based approach in Section 4.1. The TAM marker of a verb phrase is a concatenation of the suffix and postpositions which indicate the tense, aspect and modality of the verb phrase. It does not indicate the person or any other feature. For example *balalAma*, *balala* and *balalena* have the same tense(past), aspect(simple) and modality(indicative). So they have same TAM marker

la. In the next step, we tried to identify the possessive relations of a sentence. In the tag set possessive relations are denoted by "r6". To identify the "r6", we have used rule based approach. We observed that most of the errors occur due to the wrong identification of "k1" and "k2" as "pof" relation. We resolved some conflicts of "pof" relations with k1 and k2 in step 3. Finally we postprocessed the noun chunks based on their postposisions and suffix markers in step 4.

4.1 TAM based root identification

We observed that in many cases the parser is not able to identify the root of the sentence correctly. To identify the root of the sentences, we looked into sentences and tried to find out some clues. We observed that when a verb occurs with some TAM marker it becomes the root of the sentence.

Consider the Bengali Sentence "*sirila balala caluna*" which means "Siril said lets go". There are two verb chunks, *balala* and *caluna*. The correct root is identified by some rules.

By observing different sentences, we detected some TAMs of the verb phrases which uniquely identify them as root of the sentence. We prepared a priority based TAM list that contains 150 such TAMs. For each simple sentence, we find the TAM attached with the verb. If the TAM occurs in the prepared TAM list, we assign that verb as the root of the sentence. In a complex sentence more than one phrase with different TAMs can occur. Among these phrases, one will be the root of the sentence. These TAMs are arranged in the priority based TAM list in order of their priority. If two entries of the TAM list co-occur in a sentence then the higher priority TAM will become the root of the sentence.

In the above example the verb chunks *balala* and *caluna* have the TAMs *la* and *ka* respectively. *caluna* is in third formal person. *caluka* is in second person which have the same tense, aspect and modality as *caluna*. So *caluka* and *caluna* have the same TAM marker *ka*. The MaltParser identifies *caluna* as the root of the parse tree and the phrase *balala* as verb modifier. In the priority based TAM list *la* comes before *ka*. So, we changed the phrase *balala* as the root of the parse tree.

There are some more TAMs not listed in the priority list. Those TAMs can not uniquely identify the root of the tree. More study is required to use them for identification of the root of the sentences.

4.2 Genitive Marker Based Possessive Relation Identification

According to the dependency tag set, r6 indicates possessive relation. It takes *ra*, *era*, *xera* etc. genitive markers. But in many cases, karta (k1) takes the same markers. To differentiate the r6 from k1 in a sentence, we look into the case of the word along with the genitive markers. If the case of a word with genitive marker is oblique, then only that word is identified as possessive(r6).

Rule: if the case of a word with genitive or possessive marker is oblique
assign r6 for that word and attach it with the next noun or NST chunk..

For example, *rAmera Cele bAdZi jAbe.* (Ram's son will go to home). In this example, the case of the *rAmera* is oblique and so it is identified as possessive(r6). If the case of the word is direct, we do not consider that word as possessive(r6). In the example, *rAmera Kixe peyZeCe* (rAma is hungry), *rAmera* is not r6. Here rAmera is karta(k1).

4.3 Resolving "pof" misidentification

"pof" (part of) dependency relation does not take any suffix. We selected the chunks which are identified as "pof" with a verb chunk having suffix or postpositions. We changed the corresponding relation to some other relation based on the following rules.

Rule1: if the suffix is *yZa* or *we* or *e*
assign *k7p* for that word
Rule2: if the suffix is *ke*
assign *k2* for that word
Rule3: if the suffix is *tA*
assign *k1* for that word

Table 2 shows some of the cases where Rule 1 is applied. MaltParser has identified the noun phrase entries of this table as *pof*. If the suffix is *e*, *we* or *yZa* then the noun phrase dependency tag is identified as *k7p* according to the Rule 1. The postposition or TAM is not used in this rule but can be used in some other rules. In this step of postprocessing we have not modified the attachments and so is not included in this table.

Head Word	Suffix	TAM	POS	Changed relation
koNe	e	me	NN	k7p
bAWarume	e	$	NN	k7p
mAWAyZa	yZa	me	NN	k7p
anuBabe	e	me	NN	k7p
e	e	$	NN	k7p
klASarume	e	$	NN	k7p
kaWAyZa	yZa	me	NN	k7p
bAdZiwe	we	$	NN	k7p
trene	e	me	NN	k7p

Table2: List of some phrases for Rule 1.

4.4 postposition and suffix marker based rules for dependency relation identification

For morphologically rich free word order languages the postposition, suffix and TAM Markers are very important to identify the proper dependency relations (Bharati et. al., 2008). The suffix and postpositions which identifies a particular relation is identified in this phase of postprocessing. In many cases a suffix or postposition is attached with a particular relation, even though there are some exceptions. We made many rules for identification of dependency relation based on the suffixes and postpositions. Some of the rules are written below.

Rule1: if the postposition is *CAdZA*
assign *vmod* for that word
Rule2: if the suffix is *kAle*
assign *k7p* for that word
Rule3: if the postposition is *janya*
assign *rt* for that word
Rule4: if the postposition is *xiyZe*
AND the POS tag is NN
assign *vmod* for that word
Rule5: if the postposition is *xiyZe*
AND the POS tag is NST
assign *k7* for that word
Rule6: if the postposition is *Weke*
assign *k5* for that word

5 Evaluation

Training data size: 980 Sentences
Test data size: 150 Sentences
Tagset: As given in AnnCorra (Sharma et. al.)
Baseline System: MaltParser version 1.2
Evaluation Module: eval07.pl
Input Bengali Sentence: *nijera*|PRP[c-o|v-era] *klAsera*|NN[c-o|v-era] *(sAmane xiyZe)*|NST[c-d|v-0_xiyZe] *yaKana*|PRP[c-d|v-0] *se*|PRP[c-d|v-0] *(neme AsaCe)*|VM[c-$|v-Be_As+Ce] *waKana*|PRP[c-d|v-0] *GatanAtA*|NN[c-$|v-tA] *Gatala*|VM[c-$|v-la].

English Translation: *Near his own class when he was getting down then the incident take place.*

In the above example Bengali input sentence the chunks with more than one words are shown in round bracket. The POS tag of the head of each chunk is separated by pipe (|). The features namely case(c) and vibhakti or TAM(v) of the chunk which are used in postprocessing are shown near each chunk in square bracket. Here PRP – Pronoun, NN – Noun, NST – Space or time, VM – Verb. Following are the actual dependency relations between the chunks of the input sentence.

r6(nijera, klAsera)
r6(klAsera, sAmane xiyZe)
k7(sAmane xiyZe, neme AsaCe)
k7t(yaKana, neme AsaCe)
k1(se, neme AsaCe)
nmod_relc(neme AsaCe, waKana)
k7t(waKana, Gatala)
k1(GatanAtA,Gatala)
root(Gatala)

The dependency relations of the chunks of the above example as given by the MaltParser version 1.2 are shown below. Wrong relations and incorrect attachments/arguments are made bold.

r6(nijera, klAsera)
k1(klAsera, neme AsaCe)
k7p(sAmane xiyZe, neme AsaCe)
k7t(yaKana, neme AsaCe)
k1(se, neme AsaCe)
root(neme AsaCe)
k7t(waKana, Gatala)
pof(GatanAtA,Gatala)
k2(Gatala, neme AsaCe)

In the above example there are two verb chunks: *neme AsaCe* and *Gatala*. The corresponding TAMs are *Be_As+Ce* and *la*. Out of them *la* comes earlier than *Be_As+Ce* in the priority based TAM list. So, in the TAM based root identification phase of postprocessing (explained in section 4.1), we changed the dependency tag of the phrase *Gatala* as the root of the sentence. We also changed its attachment as 0. In the r6 identification phase of postprocessing (explained in section 4.2) we found the phrase *klAsera* is having case – oblique and suffix *era*. So its dependency tag is changed to r6 and attachment is changed to the next NST phrase. In the pof re-

solving phase of postprocessing (explained in section 4.3) we found the phrase *GatanAtA* is made pof. It has the suffix *tA*. So, Rule 3 is applied and its tag is changed to k1. In the postposition and suffix based postprocessing (explained in section 4.4) we found the phrase with the head *sAmane* is having the suffix *xiyZe* and its POS tag is NST. So, rule 5 is applied and its tag is changed to k7. The final relations of the above example after all the four postprocessing are given below. Here there is one incorrect relation which is made bold.

r6(nijera, klAsera)
r6(klAsera, sAmane xiyZe)
k7(sAmane xiyZe, neme AsaCe)
k7t(yaKana, neme AsaCe)
k1(se, neme AsaCe)
root(neme AsaCe)
k7t(waKana, Gatala)
k1(GatanAtA,Gatala)
root(Gatala)

The improvements of the accuracies by the postprocessing approaches are shown in the Table 3. Here we can see that the first two postprocessing phases have improved all the three scores. But last two postprocessing stages have not improved unlabeled attachment score. That is due to the fact that we have not changed the attachment in these two stages. Only dependency tag values are changed. So, only labeled scores have improved in these two stages.

	Labeled att.(%)	Unlabeled att.(%)	Label acc.(%)
Baseline	54.62	79.28	58.45
Postpro 1	56.72	80.89	60.67
Postpro 2	58.45	82.61	62.52
Postpro 3	59.93	82.61	64.00
Postpro 4	60.79	82.61	65.23

Table3: evaluation results of all stages (Postpro-postprocessing, att-attachment, acc-accuracy)

6 Conclusion

The postprocessing rules we have suggested can be applied on the output of any data driven system. We have checked MaltParser and MSTParser and by the postprocessing we have got highest accuracy in the first one. Both of these systems can not use all the features (Bharati et. al., 2008). Our rules are mainly hard constraints (Bharati et. al., PACLIC, 2009) (Bharati et. al., *IWPT09*, 2009) based on the grammatical features that in general can not be broken. If a system is built which uses all the features then the rules may become less effective.

These rules are not complete in handling all the error patterns. The rules are built based on the patterns of errors in MaltParser system output. Applying these rules on the output of some other system may not effect. More study is required to prepare a comprehensive rule set to further improve the results.

Reference

Bharati A., Husain S., Ambati B., Jain S., Sharma D. M. and Sangal R. Two semantic features make all the difference in Parsing accuracy. In *Proceedings of the 6th International Conference on Natural Language Processing (ICON-08),* CDAC Pune, India. 2008.

Bharati A., Husain S., Sharma D. M. and Sangal R. 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.

Bharati A., Husain S., Vijay M., Deepak K, Sharma D. M. and Sangal R. Constraint Based Hybrid Approach to Parsing Indian Languages. In *Proceedings of The 23rd Pacific Asia Conference on Language, Information and Computation (PACLIC 23)*. Hong Kong. 2009.

Bharati, A., R. Sangal, and T. P. Reddy. A Constraint Based Parser Using Integer Programming, In *Proceedings of ICON*. 2002.

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

Ethnologue: Languages of the World, 16th edition, Edited by M. Paul Lewis, 2009.

Karlsson, F., A. Voutilainen, J. Heikkilä and A. Anttila,(eds). Constraint Grammar: A language-independent system for parsing unrestricted text. *Mouton de Gruyter.* 1995.

Nivre J., Hall J. and Nilsson J. MaltParser: A Data-Driven Parser-Generator for Dependency Parsing. In *Proceedings of the fifth international conference on Language Resources and Evaluation (LREC2006)*, May 24-26, 2006, Genoa, Italy, pp. 2216-2219

Sharma D. M., Sangal R., Bai L. and Begam R. *AnnCorra : TreeBanks for Indian Languages (version – 1.9)*