Dependency Parser for Bengali: the JU System at ICON 2009

Aniruddha Ghosh[†] **Pinaki Bhaskar**[*] **Amitava Das**[+] **Sivaji Bandyopadhyay**[¥]

Department of Computer Science and Engineering
Jadavpur University
Jadavpur, Kolkata 700032, India
arghyaonline@gmail.com[†] pinaki.bhaskar@gmail.com[*] amitava.santu@gmail.com[+] sivaji_cse_ju@yahoo.com[¥]

Abstract

This paper reports about our work in the ICON 2009 NLP TOOLS CONTEST: Parsing. We submitted two runs for Bengali. A statistical CRF based model followed by a rule-based post-processing technique has been used. The system has been trained on the NLP TOOLS CONTEST: ICON 2009 datasets. The system demonstrated an unlabeled attachment score (UAS) of 74.09%, labeled attachment score (LAS) of 53.90% and labeled accuracy score (LS) of 61.71% respectively.

1 Introduction

Bengali language is characterized by a rich system of inflections (VIBHAKTI), derivation, and compound formation (Saha et al., 2004; Dash, 1994; Chakroborty, 2003) and karakas, which is why analysis and generation involving Bengali is a very challenging task. Parsing natural language sentences pose considerable difficulties due to the ambiguous nature of natural languages. Therefore, developing a computational grammar for a natural language is a complicated endeavor.

Most of the previous research attempts in parsing of Bengali were based on the detection and formation of the proper rule set to identify characteristics of inter-chunk relations (Sengupta, P. and B. B. Chaudhuri. 1993; Saha, 2005). Dependency based methods for syntactic parsing has become quite popular in processing of natural languages in recent years. The dependency parser developed for Bengali and presented in this report is statistical in nature and a conditional random field (CRF) model has been used for the same. A separate rule based dependency parser for Bengali has also been developed. The output of the baseline CRF based system is filtered by a rule-based post-processing module using the output obtained through the rule based dependency parser.

2 The Dependency parsing system

A parser can be defined as a language model over a word lattice in order to determine what sequence of words running along a path through the lattice has highest probability. Capturing the tree structure of a particular sentence has been seen as key to the goal of disambiguation. One way to capture the regularity of chunks over different sentences is to learn a grammar that explains the structure of the chunks one finds. The technique presented in this paper uses both statistical and rule based models to disambiguate between dependency relations. Statistical model works well to capture language model. Grammatical rules are manually developed. The crucial features for detecting dependency relations are identified and used in the training of the CRF system. The post-processing rules are manually devised by analyzing the training set.

2.1 The Statistical Parser

The probabilistic sequence models, which allow integrating uncertainty over multiple, interdependent classifications and collectively determine the most likely global assignment, may be used in a parser. A standard model, Conditional Random Field (CRF), has been used.

2.2 Conditional Random Fields (CRF)

CRFs are undirected graphical models which define a conditional distribution over a label sequence given an observation sequence. We de-

fine CRFs as conditional probability distributions $P(Y|X)$ of target language words given source language words. The probability of a particular target language word Y given source language word X is the normalized product of potential functions each of the form

$$e^{(\sum_j \lambda_j t_j(Y_{i-1},Y_i,X,i))+(\sum_k \mu_k s_k(Y_i,X,i))}$$

Where $t_j(Y_{i-1}, Y_i, X, i)$ is a transition feature function of the entire source language word and the target language characters (n-gram) at positions i and $i-1$ in the target language word; $s_k(Y_i, X, i)$ is a state feature function of the target language word at position i and the source language word; and λ_j and μ_k are parameters to be estimated from training data.

$$F_j(Y,X) = \sum_{i=1}^{n} f_j(Y_{i-1}, Y_i, X, i)$$

where each $f_j(Y_{i-1}, Y_i, X, i)$ is either a state function $s(Y_{i-1}, Y_i, X, i)$ or a transition function $t(Y_{i-1}, Y_i, X, i)$. This allows the probability of a target language word Y given a source language word X to be written as

$$P(Y|X,\lambda) = (\frac{1}{Z(X)})e^{(\sum \lambda_j F_j(Y,X))}$$

Z(X) is a normalization factor.

The parameters of the CRF are usually estimated from a fully observed training data $\{(x^{(k)}, y^{(k)})\}$. The product of the above equation over all training words, as a function of the parameters λ, is known as the likelihood, denoted by $p(\{y^{(k)}\}|\{x^{(k)}\}, \lambda)$. Maximum likelihood training chooses parameter values such that the logarithm of the likelihood, known as the log-likelihood, is maximized. For a CRF, the log-likelihood is given by

$$L(\lambda) = \sum_k [log \frac{1}{Z(x^{(k)})} + \sum_j \lambda_j F_j(y^{(k)}, x^{(k)})]$$

From the input file in the SSF format, all the morphological information like root word, chunk labels, POS tags, vibhakti, case markers are retrieved. The dependency relation of a chunk in a sentence depends on the morphological information along with the position in the sentences and also the surrounding chunk labels. Therefore the CRF statistical tool calculates the probability of the morphological information along with the dependency relation of the previous 3 and next 3 chunks. In this parser, quad-gram technique is used as most of the sentences have around 10 chunks. It can calculate the dependency among the rest of the dependency tags that has not been marked in that sentence to resolve the ambiguity because occurrence of same dependency tag in a sentence has a low probability. The system works fine for simple sentences. As many phrasal structures are possible in complex and compound sentences, CRF system is prone to make mistakes. Thus, during post processing, clauses present in such sentences are identified using the conjunctive words (ex. “ebam”) or subordinate words (ex. “se”, “ye”). Such post processing rules are based on linguistic knowledge.

The input file in the SSF format includes the POS tags, Chunk labels and morphology information. The chunk information in the input files are converted to B-I-E format so that the begin (B) / inside (I) / End (E) information for a chunk are associated as a feature with the appropriate words. The chunk tags in the B-I-E format of the chunk with which a particular chunk is related through a dependency relation are identified from the training file and noted as an input feature in the CRF based system. The corresponding relation name is also another input feature associated with the particular chunk. Each sentence is represented as a feature vector for the CRF machine learning task. After a series of experiments the following feature set is found to be performing well as a dependency clue. The input features associated with each word in the training set are the root word, pos tag, chunk tag, vibhakti and dependent chunk tag and dependency relation.

Root Word: Some dependency relations are difficult to identify without the word itself. It is better to come with some example.

```
AjakAla  NN   NP   X    k7t
```

In the example there is no clue except the word itself. The word itself is noun, the chunk level denotes a noun phrase and there is no vibhakti attached to the word. For these cases, word lists of temporal words, locations names and person names have been used for disambiguation (Ekbal et. al., 2008). Specifically identification of k7t relation is very tough because the word itself will be a common noun or a proper noun but the information of whether the word denotes a time or a location helps in the disambiguation.

Part of Speech: Part of speech of a word always plays a crucial role to identify dependency relation. For example dependency relations like k1 and k2 in most of the cases involve a noun. It has been observed through experiments that not only POS tag of present word but POS tags of the context words (previous and next) are useful in

identifying the dependency relation in which a word takes part.

Chunk label: Chunk label is the smallest accountable unit for detection dependency relations and it is an important feature. But sentences are parsed into word level during training, hence chunk labels are associated to the appropriate words with the labels as B-X (beginning), I-X (Intermediate) and E-X (End) (where X is the chunk label).

Vibhakti: Indian languages are mostly non-configurational and highly inflectional. Grammatical functions (GFs) are predicted by case inflections (markers) on the head nouns of noun phrases (NPs) and postpositional particles in postpositional phrases (PPs). In the following example the '0_janya' vibhakti inflection of the word "pAoyZAra" leads to *rh (*Hetu*)* case inflections. However, in many cases the mapping from case marker to GF is not one-to-one. Some example words and the corresponding vibhaktis are shown in Table 1. The classical technique for non-configurational syntactic encoding of GFs (Bresnan, 1982) therefore requires a number of alternations to be thrown in to handle this phenomenon.

Word	Vibhakti
bAMlAyZa	Null
ema e	Null
padZawe	A_As+Ce
ekatA digrI	0
pAoyZAra.	0_janya

Table 1: Words and associated Vibhakti

The output of this statistical parsing/mapping process is a dependency relation along with the root word, vibhakti, chunk tag and their corresponding dependency relation with the chunk heads. During the development process confusion matrix helped to detect errors. The most prominent errors on the development set when vmod was wrongly identified are shown in Table 2.

	Vmod
k1	42
k2	37
k7p	14
Ccof	41

Table 2: Confusion Matrix on Development Set

2.3 Post-Processing

With the help of confusion matrix, we have devised a set of parsing rules depending on the nature of errors to disambiguate when more than one dependency relation has been identified in a certain situation.

A rule based dependency parser that uses linguistic knowledge has been developed to check the output of the CRF based dependency parser. Depending on specific attributes of the chunk like vibhakti/case markers and/or word information, the rule based system derives the dependency relation of the chunk. For each dependency relation tag depending on specific linguistic features, syntactic cues are derived to identify the dependency relations. Some example rules used are described below:

1. A NP chunk with 0 vibhakti and NNP or PRP postag will be marked with k1 relation with the nearest verb chunk.
2. A chunk with "era" vibhakti will be marked with 'r6' relation with next noun chunk.
3. A NP chunk with 0 vibhakti and NN postag will be marked with k2 relation with the nearest verb chunk.
4. In co-ordinate type sentences, the verb chunk will be marked with 'ccof' relation with the nearest CCP chunk. If CCP chunk is surrounded by two NP chunks then both the NP chunks will be marked as 'ccof' with the CCP.
5. In sub-ordinate type sentences, the verb chunk of sub-ordinate clause will be marked with "nmod__relc" with that chunk modified by the main clause.
6. If a chunk marked with "0_weke", k5 relation will be identified.
7. If a chunk marked with "0_prawi", 'rd' relation will be indentified. The relations 'k5' and 'rd' are pre-dependent, i.e., a dependent that precedes its head.
8. Some verbs like "kar" or "ha" etc, expects arguments. For these verbs the previous chunk will be expected argument of the verb chunk. The previous chunk will be identified as "pof" relation with the following verb. For example:

```
(( Suru NN ))_NP (( hayZe VM
| SYM ))_VGF
```

The “ha” verb expects argument. Here “Suru” is the expected argument. So the NP chunk is identified as “pof” relation with the verb chunk.

```
(( apekRA NN ))_NP (( karawe
   VM ))_VGF
```

The NP chunk preceding the “kar” verb is marked with “pof” relation with the VGF.

The ambiguity comes when for a certain vibhakti, multiple possible relations are identified. For example, for a chunk with “0” vibhakti two possible output dependency relations are ‘k1’ & ‘k2’. This ambiguity is resolved using pos tag information. If pos tag is ‘NNP’ then dependency relation will be ‘k1’ and if ‘NN’ then it will be ‘k2’. If ambiguity is not resolved with this rule then position of the chunk in the sentence is considered. If there are two chunks with “0” vibhakti, the distant chunk from the verb chunk will be marked with ‘k1’ relation and nearer ones will be marked with ‘k2’ relation.

Sometimes to resolve the ambiguity, system checks the categorization of the root word to identify the relation. For example:

The vibhakti “me” identifies multiple relations e.g., k7, k7p, k7t, k3. This ambiguity can be resolved by examining the root word category and case marker. If it is a time related word (e.g. kAl, waKana etc.) then relation will be k7t. If it is a location word then it will be k7p. In root word, if the case marker is ‘d’ then the relation will be k3, otherwise k7 will be marked.

The outputs of the CRF based and the rule based dependency parsers are matched. The rule based system is given the higher priority as it is based on syntactic-semantic cues. If there is any mismatch between the two results then output of rule based system will be taken.

3 Error Analysis

During the development stage of the system we had studied the various chunk labeling errors committed by the system.

```
(( wAra PRP ))_NP (( Ay-
Zawana NN ))_NP (( o CC
))_CCP (( parimANa NN
))_NP (( xeKe VM ))_VGNF
(( buJawe VM ))_VGNF ((
asubiXA NN ))_NP (( hay-
Za VM ))_VGF (( ei DEM
paWa NN ))_NP (( hAwi
NN ))_NP (( geCe VM à¥¤
SYM ))_VGF
```

In the above example, it is very much ambiguous to derive the syntactic cues for the case inflection of the NP with the word “ei paWa” depending on the syntactic and morphological information. In dependency parsing, morphological information plays a crucial role to identify case inflection. Deficiency of morphological information leads to ambiguous identification. As an example:

```
(( NP
<af=kOwUhala,unk,,,,,, >
SYM
<af=",punc,,,,,,>
anAbaSyaka JJ
<af=anAbaSyaka,adj,,,,,,>
kOwUhala NN
<af=kOwUhala,unk,,,,,,>
))
```

In the above example, using morphological information no syntactic cues could be formed. Even other linguistic and syntactic (morph-syntactic) information like POS tag, Chunk tag, root word will not lead to an unambiguous dependency relation.

3.1 Experimental results

We have trained the probabilistic sequence model with the morphological features like root word, POS-tag, chunk tag, vibhakti, dependency relation from the training set data. A brief statistics of the datasets are presented below. In the training set, out of 980 sentences, 223 sentences are simple sentences, 757 compound sentences. Among the compound sentences 198 are co-ordinate sentences and 559 are sub-ordinate sentences. The accuracy of the CRF based model on the development set is 65%. Depending upon the nature of errors involved in the results, we have devised the template with new rules. The use of the new template in the CRF tool increased the accuracy up to 69%.

With this result of CRF based model, rule based model’s output is merged. The rule based model output has a higher priority as it is based on syntactico-semantic attributes of the chunk. This merging with rule based system increases the accuracy to 74%. The accuracy figures of the present Bengali parser on the test set are UAS (Unlabelled Accuracy Score) is 74.09%, LAS (Labeled Accuracy Score) is 53.90% and LS (Labeled Score) is 61.71% respectively.

4 Conclusion

This paper reports about our works as part of the NLP Tool Contest at ICON 2009. We have used the statistical CRF based model along with rule based post-processing. During the CRF based run, we have obtained the accuracy of 69% in the second run. We have used a rule-based post-processing and produce a better accuracy value of 74%.

A properly designed NLP platform for Indian languages must come with an efficient morpho-syntactic unit for parsing words into their constituent morphemes where lexical projections of words can be obtained from projections of individual morphemes. Phrasal order could be vary depending on the corpus. In future task our aim is to develop a more proficient statistical system, can produce more than one possible parsed output. A more concise rule set should be generated with morpho-syntactic and lexico-syntactic variations.

References

Asif Ekbal, Rejwanul Haque, Amitava Das, Venkateswarlu Poka and Sivaji Bandyopadhyay. 2008. *Language Independent Named Entity Recognition in Indian Languages.* In Proceedings of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages. pp. 33-40.

Bresnan, J. 1982. *Control and Complementation.* In J. Bresnan, editors, The Mental Representation of Grammatical Relations. MIT Press, Cambridge, MA, pp. 282-390.

Chakroborty, B. 2003. *Uchchotoro Bangla Byakaron*. AkshayMalancha.

Dash, K.C.(Ed.). 1994. *Indian Semantics*. Agamakala Publications,Delhi.

Saha, G.K., Saha, A.B., Debnath, S. 2004. *Computer Assisted Bangla POS Tagging*. iSTRAN, Tata McGraw-Hill, NewDelhi.

Saha, Goutam Kumar. 2005. *English to Bangla Translator: The BANGANUBAD*. International Journal-CPOL, Vol. 18 (4), pp.281-290, USA.

Sengupta, P. and B. B. Chaudhuri. 1993. *A Morpho-Syntactic Analysis Based Lexical Sub-System*. International Journal of Pattern Recognition and Artificial Intelligence. pp595--619.