The Dhaka University Journal of Linguistics : Vol. 10, No. 20, Published by Registrar, University of Dhaka, ISSN 2075-3098

Developing Digital Resources for Computational Bangla

Naira Khan*

Abstract : As the world moves towards a digitally-literate global society, digitising languages has become integral for information exchange in every language. Despite being one of the most widely spoken languages of the world, Bangla is one of the most digitally under-resourced languages. In this respect, Bangla computing has become an essential next phase in the evolutionary path of the language. A number of endeavours in computational modeling can be noted as setting the precursors for a robust repository of computational resources for Bangla. From corpus-development, to Bangla WordNet, to POS tagging, this paper conducts a survey of the state of Bangla computing as undertaken in the form of various projects, and provides an overview in the progress being made in developing respective digital resources.

1. Introduction

With the increase of technology and computerisation, computational modeling of Bangla (exonym: Bengali) is fast becoming necessary. Although Bangla is the 5th most widely spoken language in the world, it is one of the most under-resourced languages in terms of digitisation. Some efforts of computational modeling can be noted as setting the precursors for a robust repository of computational resources for Bangla. With a brief introduction to the Bangla language (§1.1) and its script (§1.2), in the following sections I conduct a survey of the state of computational modeling in Bangla and other digital resources that have been undertaken in the form of projects.

1.1 The Bangla Language

1.1.1 Origin and History

Bangla is said to belong to the Satem branch of the Indo-European language family. The Satem branch evolved into Indo-Iranian and branched off into Indic or Indo-Aryan and Iraninan. Bangla originated

* Assistant Professor, Department of Linguistics, University of Dhaka

from the Eastern branch of Indo-Aryan, which descended from Indo-Iranian but separated from Iranian (Ethnologue 2005).

1.2.2 Development

The Indo-Aryan branch underwent three stages of development:

a. Old Indo-Aryan: Vedic
b. Middle Indo-Aryan: The Prakrit
c. New Indo-Aryan : Bengali, Hindi, Assamese, Gujrati etc.

The Bangla language itself underwent three stages of development:

a. Old Bangla (900/1000–1400)—texts include *Charyapada* devotional songs;
b. Middle Bangla (1400–1800)—major texts of the period include Chandidas's *Srikrishnakirtan*;
c. Modern Bangla (since 1800)—Marked by the diglossic Sadhu and Chalit variety.

(Shahidullah, 2002)

1.2.2.1 Standard Colloquial Bangla (SCB)

From the 1800s, Modern Bangla evolved until Chalit slowly displaced Sadhu. The diglossic situation of Sadhu and Chalit gradually diminished with the emergence of a standard form comprising the variety used by the educated elite of Kolkata, India with Rarh as its hinterland. (Chatterjee, 2002)

Over the course of time SCB has diverged into two recognizable standard varieties of Bangla with marked differences in pronunciation and lexical usage:

i) The Kolkata Standard: The standard form used in the Indian State of West Bengal
ii) The Dhaka Standard: The standard form used in Bangladesh.

1.2 Speaking population and geographical distribution of Bangla

The Bangla language is spoken by 189,000,000 (appx) and is the state language in Bangladesh as well as the Indian State of West Bengal (Ethnologue, 2016). It is also spoken in Malawi, Nepal, Saudi Arabia, Singapore, United Arab Emirates, United Kingdom, and USA. In terms of number of speakers it is said be the 5th largest language in the world (Ethnologue, 2016).

1.3 Script Background

A descendant of the Brahmi script and used to write Modern Bangla, the Bangla script evolved from the Proto-Bangla scripts of the Northern type. Its closest relatives are the Assamese, Maithili, Oriya scripts as well as Manipuri and the Newari scripts, and apart from a few minor typographical differences these scripts are practically identical (Masica, 2001). The Bangla script is closely related to the Devanagari script but visibly different in its markedly angular shape, which is said to have resulted from differences in primitive writing implements that were used to record the language at the time. As part of the legacy of Brahmi the Bangla script is an abugida script i.e. an alphasyllabary with the inherited the characteristics of:

i) The inherent vowel

ii) The attached sign for other vowels

iii) The compound ligature for consonant clusters

In the Bangla script, each consonant has a high back rounded vowel as an inherent vowel. The inherent vowel is deleted when additional vowel signs are added in the form of dependent diacritics ('kar' signs). The script also inherited the phonetic arrangement of letters i.e. consonants are not arranged arbitrarily but rather according to phonetic features (Masica, 2001). The letters representing Bangla consonants and vowels are given below:

i) Bangla consonantal letters:

ক খ গ ঘ ঙ (velars)
চ ছ জ ঝ ঞ (palatals)
ত থ দ ধ ন (dentals)
ট ঠ ড ঢ ণ (post-alveolars)
প ফ ব ভ ম (bilabials)
য র ল য় (approximants)
স শ ষ হ (fricatives)
ড় ঢ় (flaps)
ৎ ং ঃ ঁ (alphabetic characters)

ii) Bangla vocalic Letters:

অ আ ই ঈ উ ঊ ঋ এ ঐ ও ঔ

It should be noted that as computerization of Bangla often deals with the language in a device setting such as a computer or a phone, more often than not computational Bangla deals with the script or written Bangla rather than spoken Bangla. In the following sections we will look at efforts of building digital resources for computational Bangla, from building corpora to parts of speech tagging (§2 - §5).

2. Corpus

2.1 Description

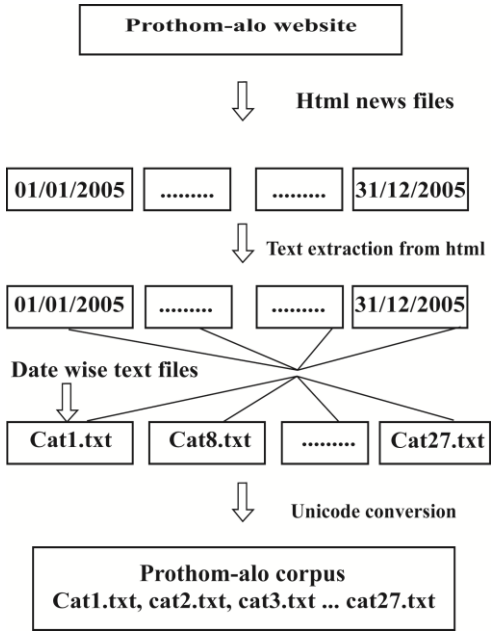
A trained annotated corpus can be a key resource to any linguistic research endeavor. In fact many intuitive studies are challenged and proven wrong on the basis of corpus analysis. However there still exists no complete corpus for the Bangla language per se. Some sporadic attempts at building both spoken and written corpora can be found however in comparison to the large corpora that exist in languages such as English, Chinese, Spanish etc. Bangla is still at an embryo level. The first project that is known to have initiated Bangla corpus generation was conducted from 1991 to 1995 by the Department of Electronics (DOE), Govt. of India and Central Institute of Indian Languages (CIIL) (Arafat, Islam and Khan, 2006). For quite some time this corpus of 3 million words has been providing most of the data to linguists working with Bangla. In fact N.S. Dash's "Corpus Linguistics and Technology", a warehouse of corpus related studies is supported by data from the CIIL corpus (Arafat, Islam and Khan, 2006). However, with the deviation of the two standards for Bangla, namely the Kolkata Standard and the Dhaka Standard, with marked differences in features there arose a need to compile a corpus for Bangladeshi Bangla i.e. Dhaka Standard. This necessity was sublimated into an endeavor by the Center for Research on Bangla Language Processing (CRBLP) to compile the 'Prothom Alo' news corpus based on 'Prothom Alo' - the most widely circulated Bengali daily in Bangladesh. Although ideally the initiative was to develop a balanced corpus however this was met with two obstacles: a dearth of available digital text in Bangla and the absence of a fully functional Bangla OCR. As a result the focus of the corpus was automatically concentrated on the available resources determining the nature of the corpus. Hence the 'Prothom Alo' corpus is primarily a new corpus and comprises 18,067,470 (eighteen million plus) word tokens and 386, 639 distinct word types (Arafat, Islam and Khan, 2006 : 2).

An unpublished speech corpus of Standard Colloquial Bangla was developed as a by product of the Text to Speech (TTS) project of CRBLP. The said corpus comprises: 13 hours 32 minutes 44 seconds of audio duration with 1068860 total words in text, 18029 unique words and 10895 total sentences. The speech corpus is phonetically transcribed and tagged on a sentence level (Alom 2008 : 4).

The 'Prothom Alo' corpus was complied in two phases: collection of raw text from the Prothom Alo website and the conversion to Unicode (Arafat, Islam and Khan, 2006).

2.1.1 Collection of Text

The raw text for the corpus was collected by using a web crawler program that surfed the 'Prothom Alo' webpage (www.prothom-alo.com) and downloaded the available news text including that from magazines and periodicals in HTML format. The text was extracted from the HTML files using Linux shell scripts with library reference to Lynx. For the benefit of research text was then categorized according to news categories. The resultant corpus is three hundred and eighteen megabytes in size (Arafat, Islam and Khan, 2006). Shown diagrammatically below:



Soursce: Arafat, Islam and Khan, 2006

2.2 Encoding Conversion

Conversion into Unicode was required due to the following problems (Arafat, Islam and Khan, 2006):

ii) Glyphs and Fonts
iii) Keyboard mapping

2.2.1 Font conversion

Prothom-Alo uses three fonts, namely "Bangsee Alpona" and "Prothoma", "Alo" both true type fonts (TTF), for the online version of the newspaper ("Prothom-Alo" www.prothom-alo.com). The encoding of all these fonts are maintained in different files, which are later required. Two font specific converters developed in Java named CRBLP converter were used to convert the collected texts files to Unicode texts (Arafat, Islam and Khan, 2006).

The CRBLP converter is a system which can convert Bangla ASCII encoded text to Unicode. There are thousands of Bangla documents written in the ASCII encoding system. ASCII encoded Bangla font and keyboard was developed before Unicode become prevalent. Automatic Unicode converter is the only solution to enter into the Unicode world. Since several million of Bengali documents, websites etc. have been written in ASCII, hence the easiest solution is an automatic Unicode converter. The difficulties faced in terms of conversion into Unicode was primarily twofold: different fonts that use different coding systems and the same font uses different coding systems in different versions. Here the converter was focused on Suttony MJ font developed by Bijoy version 2000 as well as Bangsee Alpona, Prothoma and Alo developed by Prothom-Alo (Alom, 2009).

The converter engine handles one-to-one, one-to-many mapping tables and exceptions. Unicode defines vowels, vowel kars, consonants, digits and symbols. For handling the conjuncts and ordering of glyph, i.e substituting, positioning of glyphs is part of the open type font (Microsoft, 2002) and USP10.dll file. The word আমি is written as আ+ম+ি maintaining congruity with oral spelling rules. This reordering is handled by USP10.dll. In an ASCII based system it is typed as আ+ি+ম. All glyphs of any true type font are separated in four forms: pre char, post char, base char and sa char (conjugate char).

2.2.1.1 Methodology

The methodology of the system is shown below.

2.2.1.2 File type identifier

The first phase is to detect the document type. The following tools are included with the converter for extracting.

- MS word text - jakarta Apache project POI
- Html text extracting - html parser
- Text file

2.2.1.3 Converter Engine

The converter engine has a list of possible exceptions and different forms of glyphs. Four possible forms of glyphs are identified for each font. For each character of input text it is determined which list the character matches and then it is converted.

2.2.1.4 Pre-Process

All glyphs of the ASCII encoded font are separated into four categories:

base-char: All full form characters.

pre-char: Character that attaches before the character it modifies.

post-char: Character that attaches after the character it modifies.

sa-char (conjunct char): All ligatures

2.2.1.5 Exceptions

The followings are identified as an exception:

1. Each of these [ঁ, গু, শু, হু, ত্ত, র্ম, রূ, হ্ম] glyphs are checked individually then converted to the corresponding Unicode number.
2. র্ (reph) will not appear after া, ী, ু, ূ, ৃ, ৌ but reph may appear after base char + া, ী, ু, ূ, ৃ, ৌ + র্
3. will appear after these char [া, ী, ু, ূ, ৃ, ৌ]
4. ঁ (chandrabindu) will always be on top of a vowel.
5. Gyph অ + া will be replaced by আ. No আ glyph available in ASCII font.

6. Halanta will be replaced by halanta + non-joiner
7. Normalization is done for ে + া -> ো and ে + ৗ -> ৌ.

2.2.1.6 Conversion process

The conversion process is handled for four different forms. For each forms the following glyph is reordered for Bangla Unicode. Such glyphs are ে ৈ ি.

The following algorithm works for each glyph of the word.

2.2.1.7 Algorithm

Step1: If the current glyph is the base char then use direct Unicode like one to one mapping.

Step2: If current glyph is pre char then use pre char + halanta unicode.

Step3: If current glyph is post char then use halanta + post char unicode.

Step4: If current glyph is sa char (conjugate) then use 1st char + halanta + 2nd char.

Step5: If current glyph is exceptions then handle it by looking up exceptions rule. Such as, reorder র্ as it appears before consonants.

Step6: If current glyph is [ি, ে, ৈ] then reorder this char after next complete char.

2.2.1.8 Results

Currently the system accuracy is above 90% (Alom, 2009). There are two possible types of error: identifying the type of glyph of the font and user data input error. User text may have wrong input which leads to wrong output. Such errors are চাঁদ -> চাঁদ. The candrabindu always appears after the vowel, however users typed it in before the vowel in ASCII. (Alom, 2009)

2.3 Corpus Processing and Statistical Analyses

Some examples of tests with the Prothom Alo corpus are given below with a comparison with CIIL corpus (Arafat, Islam and Khan, 2006):

Top ten most frequent words in the Prothom-Alo corpus

Word	Percentage	Word	Percentage
ও	1.23%	হয়	0.57%
এ	0.92%	করা	0.52%
করে	.084%	তার	0.49%
না	.072%	এবং	0.46%
থেকে	0.62%	হয়েছে	0.43%

Source : Arafat, Islam and Khan, 2006

Top ten most frequent words in the CIIL corpus

Word	Percentage	Word	Percentage
না	1.15%	এবং	0.65%
করে	0.99%	এই	0.65%
এই	0.94%	থেকে	0.55%
ও	0.91%	আর	0.51%
হয়	0.76%	তার	0.5%

Source : Arafat, Islam and Khan, 2006

Although the 'Prothom Alo' yields interesting results it is still a focused corpus in that it is based primarily on printed news. However it provides a valuable starting point in building a comprehensive corpus for Bangladeshi Bangla where previously there existed none.

3. Lexicon

3.1 Linguistic Analysis and Design

A lexicon is an essential language resource. It is the central repository of data for all language processing applications. A proposal was made to build a lexicon based on a collaborative effort through stand alone application and web based interface (Dewan, Sarkar and Khan, 2006). The words in the lexicon would be annotated with a combination of

tags addressing parts-of-speech, syntactic, semantic and other grammatical features. This endeavor aimed to provide an integrated user – friendly software interface to the user to annotate a large existing Bangla word set and proposed a mechanism to collaboratively integrate linguists and other interested people into the lexicon build up process (Dewan, Sarkar and Khan. 2006).

Due to the unavailability of a complete Bangla corpus the process of automatic lexicon development did not go too far. Hence the proposal was to manually build up a lexicon and tag the words with features such as word meaning, Parts-Of-Speech (POS) and other grammatical features. The aim of the project was to formalize a procedure for a collaborative effort by different individuals or groups towards producing a tagged Bengali lexicon. This requires a POS tagging interface, both web based and stand alone that would provide a common platform for different contributors to enter tag information, semantic and other grammatical information that is available in a dictionary (Dewan, Sarkar and Khan. 2006).

3.2 Current Work

3.2.1 Lexica

There are three types of lexica that have been manually compiled for various purposes by CRBLP (Alom, 2009):

a. A pronunciation lexicon with 92,567 ~93K entries comprising

- lexical entries
- pronunciation in the Bengali
- script
- pronunciation in IPA

b. A lexicon manually tagged for POS comprising 144833 entries with 49371 nouns, 63790 verbs, 30099 adjectives, 248 pronouns and 1325 indeclinables (adverbs, conjunctions etc.).

The second lexicon was later augmented to include:

- Registers and terminologies
- Bangla to English dictionary
- Bangla to Bengali dictionary

c. The CRBLP 'Prothom Alo' lexicon created as a byproduct of the 'Prothom Alo' Corpus comprising

3.2.2 Pronunciation

Apart from the pronunciation lexicon there has been sparse work on pronunciation at CRBLP. This work has had primarily two outputs:

1. A database of Bangla phonemes and diphones (without phonotactic sorting) created from the speech corpus (Alom, 2009).

2. A rule based automated pronunciation generator for Bengali (Mosaddeque, UzZaman and Khan, 2006).

The latter is an APG that takes a Bangla word and generates the pronunciation through grapheme to phoneme mapping. The Bangla script is not completely phonetic since not every word is pronounced according to its spelling. Therefore, it was necessary to use some pre-defined rules to handle the general cases and some case specific rules to handle exceptions. There are some rules that have been developed by observing general patterns, e.g., if the length of the word is three full graphemes then the inherent vowel of the medial grapheme (without any vocalic allograph) tends to be pronounced as [o], provided the final grapheme is devoid of vocalic. When the final grapheme has adjoining vocalic allographs, the inherent vowel of the medial grapheme tends to be silent. In the web version of the APG, queries are taken in Bengali text and it generates the phonetic form of the given word using IPA transcriptions. Furthermore, there is another version of the system which takes a corpus (a text file) as input and outputs another file containing the input words tagged with the corresponding pronunciations. The actual challenge in implementing the APG was dealing with polyphone graphemes. Due to the lack of a balanced corpus, it was necessary to adopt a rule-based approach for developing the APG. However, a possible future upgrade would be implementing a hybrid approach comprising both a rule based and a statistical grapheme-to-phoneme converter (Mosaddeque, UzZaman and Khan, 2006).

3.2.3 Morphology

Another project that was undertaken at CRBLP parallel to developing the lexicon was the morphological analyzer for Bangla called JKimmo – a multilingual morphological open-source framework that uses the PC-

KIMMO two-level morphological processor and provides a localized interface for Bengali morphological analysis. (Islam and Khan, 2006)

The goal was to create a robust and reusable framework for morphological analysis of Bengali with three primary components: the generative morphological rules, the underlying morphological processor, and the computational interface through which the user experiments with language morphology. There has been previous work in developing computational morphology for Bangla, using both simple rewriting rules and feature unification grammars (Sengupta and Chaudhuri, 1996) (Bhattacharya et al., 2005) (Dasgupta and Khan, 2004) (Dasgupta and Khan, 2004). An effort to create an interface for Bangla morphological analysis has been developed at the Indian Institute of Technology–Kharagpur (Bangla, www.mla.iitkgp.ernet.in/morph_analyzer.html), which provides a web interface to the underlying morphological engine using the iTRANS transliteration scheme. Another such effort is the Xerox Arabic Morphological Analyzer and Generator (Beesley, 2001), created with the Xerox Finite-State Technology. The Xerox system accepts modern standard Arabic words and returns morphological analyses and glosses. It has a Java Applet interface and uses ISO-8859-6 and Unicode character encodings. It is notable that none of these systems, unlike Jkimmo, is easily extendible to other languages using Unicode-encoded input and output. PC-KIMMO is one of the most widely available two-level morphological analyzer that implements Kimmo Koskenniemi's two-level morphology. The missing framework required to handle complex scripts was provided by the Jkimmo. The implementation uses Java Native Interface [10] as the bridge between PC-KIMMO and the Unicode-enabled user interface, allowing the user to experiment in any script supported by the Unicode standard. Since the analysis framework uses standard internationalization (i18n) schemes, it is trivially localized to any language by using property files for interface definitions, and transliteration schemes for the Latin-Unicode-Latin conversion needed to interface to PC-KIMMO backend. (Islam and Khan. 2006)

3.3 Wordnet

To create a WordNet for a new language is a significant challenge, not the least of which is the availability of the lexical data, followed by the

software framework to build and manage the data. BWN is a software framework to build and maintain a Bangla Word Net.

The basic building block of WordNet is a synonym set or Synset, a word sense with which one or more synonymous words or phrases are associated. Each synset in WordNet is linked with other synsets through the lexical relations synonymy and antonymy, and the semantic relations hypernymy, hyponymy, meronymy, troponymy, etc. The applications of WordNet range from creating digital lexica to performing word-sense disambiguation in automatic machine translation. The synonym set {পাখি, গগণগতি, খেচর, চিড়িয়া, নভৌকা, পংখি, পঙ্খী, পক্ষধর, পক্ষালু, পক্ষী, পতগ, পত্রী, বিহগ, বিহঙ্গ} and {পাখি, জমির একক বিশেষ, ৩০ কানি ভূমি, ২৬-৩৩-৩৫ শতাংশ, অঞ্চল একক] for example can serve as an unambiguous differentiator of the two meanings of "পাখ" (Faruqe and Khan, 2008). Such synsets are the basic entities in WordNet; each sense of a word is mapped to a separate synset in the WordNet, and synset nodes are linked by semantic relationships, such as hypernymy. The primary focus of Bengali WordNet (BWN) is on design and implementation – a framework to enable building and using Bengali WordNet (Faruqe and Khan, 2008).

The design of Bengali WordNet closely follows that of the English Word Net (wordnet sql builder, website http://wnsqlbuilder. sourceforge.net/ schema.html). The software design of Bangla Word Net allows linguists to import lexical data in a "batch" mode, as well as visual querying and editing of all the relations in the data. The basic design to support data import and subsequent queries is relatively simple; however, support for incrementally building the WordNet and for editing the data using a visual interface are two key features of BWN, and these complicate the design in a significant way. (Faruqe and Khan, 2008)

There are two common approaches for building a WordNet for a target language: (i) a top-down approach, using an existing WordNet in a source language to seed the linguistic data for the target language WordNet, and (ii) a bottom-up approach, where linguists create the WordNet synsets from scratch without depending on an existing one. The first approach has been tried for a number of WordNets (Sinha, Reddy and Bhattacharyya, 2006) (Vossen, 1999) (Farreres et al., 1998) (Barbu, Eduard and Mititelu, 2005) (Chakrabarti, Rane and Bhattacharyya, 2004). There have been many other recent attempts at

building a WordNet quickly, such as creating lexical networks by using the web or some well-structured corpora such as Wikipedia, or the BNC corpus. All of these require linguistic resources not yet available for Bengali, hence the bottom-up approach was the most practical one – by starting with the words in the target language instead of using an existing WordNet. For BWN, the starting point was translating the ontology, and words were chosen using a frequency list from the Prothom Alo corpus. These synsets were compiled in lexical source files, which were then injected into the WordNet database using a "grinder", and the resulting system can then be used through a set of interfaces. (Faruqe and Khan, 2008)

A WordNet software system comprises four parts (Faruqe and Khan, 2008):

- Lexical Source Files: These files contain the synsets that are manually compiled by the lexicographers, and are used to eventually populate the WordNet database. The schema used for nouns in the lexical source file is shown below:

  Word name/Word name (english) / description /

  Pos/||*description(english)*||

  Hypernyms:

  Synonyms:

  And a sample "noun" record is shown below.

  কাজ work|কিছু করা বা তৈরির লক্ষ্যে সরাসরি কার্যক্রম | বিশেষ্য |

  ||work -- (activity directed toward making or doing something)||

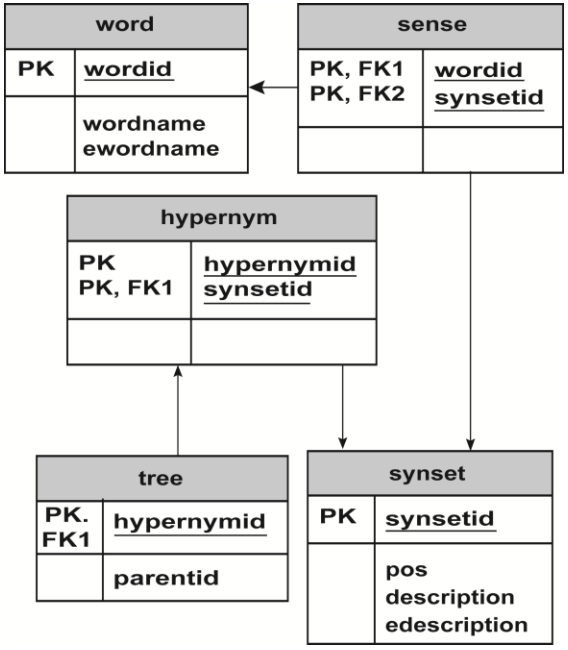
  hypernyms:| কার্যক্রম | কৃতকর্ম | ঘটিত বিষয় | মনস্তাত্ত্বিক-বিষয় | বিমূর্তন | বিমূর্ত-সত্তা | সত্তা |

  synonyms: কর্ম, কর্মকা ঁ, কাজ, কাজকাম, কাম, কার্য

  Grinder: The grinder is used to convert the lexical source files in a form that can be injected into the WordNet Database (WNDB). Basically, it parses and processes the text from the lexical source files into records, and then stores each record in the WNDB.

- WordNet Database (WNDB): WNDB is the heart of WordNet for any language. For BWN, the basic design is similar to "Wordnet SQL Builder" (wordnet sql builder, website http://wnsqlbuilder.

sourceforge.net/schema.html). However, there are significant differences under the hood, primarily to support incremental building of the database, and editing of the synsets directly via the user interface. One of the design goals is to ensure that WNDB is extensible to new lexical relations between synsets. In addition, in the word table, the English word is stored so that it can be used to link to other WordNets such as the EuroWordNet in the future. In the sense table, both the word and the synset are mapped together. In the synset table, an ID is generated for a synset but does not create the synset itself. The synset is regenerated at run-time from the sense and word tables, which plays a very important role in the case of an edit or update operation.



Block diagram of the WordNet database

Source: Faruqe and Khan, 2008)

- WNDB Interface: There are essentially three different interactions with WNDB:
- To create the initial database using the lexical source files, and then to incrementally update the database, which is a feature that significantly contributes to the database schema complexity;

- To use the database to query the data; and,
- To edit the lexical data, which is the other reason behind the database schema complexity.

These interactions are described in more details below (Faruqe and Khan, 2008):

1. *Update WNDB*: The Grinder takes each record from the lexical source file, splits the text according to the database field and then stores it into the database. The process is illustrated with the following sample record:

   "কাজ |work| কিছু করা বা তৈরির লক্ষ্যে সরাসরি কার্যক্রম | বিশেষ্য |

   ||work -- (activity directed toward making or doing something)||

   hypernyms:| কার্যক্রম | কৃতকর্ম| ঘটিত বিষয় | মনস্তাত্ত্বিক-বিষয় | বিমূর্তন | বিমূর্ত-সত্তা | সত্তা | synonyms: কর্ম, কাজকর্ম"

   After splitting the text, the grinder updates the word table with the value of *wordid* (auto incremented integer), *wordname*, and *ewordname*. Each synonym word is also entered into the word table.

   The Grinder then updates the synset table with *synsetid* (auto incremented integer), *description*, *edescription*, and *pos*.

   The Grinder then updates the sense table with those *wordids* and the particular *synsetid.*

   To update the hypernym table, we need the *synsetid* of that particular record and its corresponding *hypernymid*; because each synset, with the exception of "entity/সত্তা", may have one or more hypernyms. For that, we have to match each hypernym with the *wordname* field's value in the word table and then take the *wordid*; with this *wordid*, we have to find out the *synsetid* (because the *hypernymid* is nothing but a *synsetid*) from the sense table. Here we assume that all of these hypernym words already exist in the word table. The tree table keeps track of the parent of each hypernym word, because hypernymy relates each child to its parent. Then the Grinder updates the *tree* table with *hypernymid* and *parentid* (which is also a *synsetid*). Since "entity/সত্তা" does not have a parent, its *parentid* is given a value of 0 (zero) to indicate that.

2) Using WNDB: The second interface to the WNDB is for querying the data in WNDB. A typical scenario is given below:

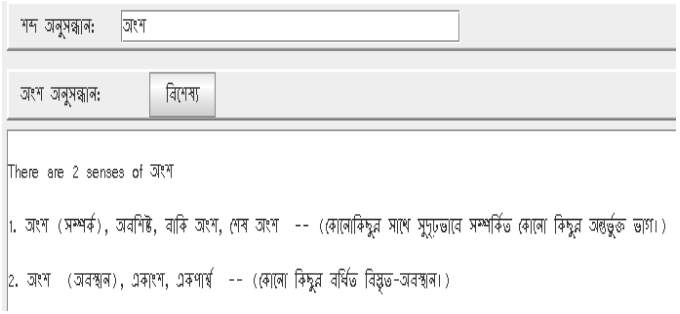


Diagram : A typical query in WNDB

Result of a search option

iv) User enters the query text into query field as shown in the following figure.

Diagram : A typical text example

v) The WNDB search engine first finds the sense (or senses) of that given word from word table then maps the *wordid* to the *synsetid* from the sense table, and then returns those *synsetids.*

In this example, "অংশ" has two senses (each word represents a single value, as mentioned earlier).

vi) For each of the resulting *synsetids*, we have to find all the *wordids* from the sense table. To create a synonym set, we have to find all the *wordnames* from the word table after matching the *wordids* for a particular synsetid. Here, we consider only one *synsetid*. For this *synsetid*, the synonyms are {অংশ (সম্পর্ক), অবশিষ্ট, বাকি}.

vii) Then, we find the description for each *synsetid* from the synset table with those *synsetids*. Then the search result is shown in the previous screenshot.

viii) To view a noun's hypernymy relation, as shown below, the application execute steps 2-4 for each sense, and then, within each sense, it performs the following steps:

ix) It finds the *hypernymids* from the hypernym table for the specific *synsetid.*

x) The application also has to track each of the hypernym's parent from the tree table to track the child-parent relation.

xi) After performing steps 5 (a) and (b), it shows the hypernym from child to parent order.

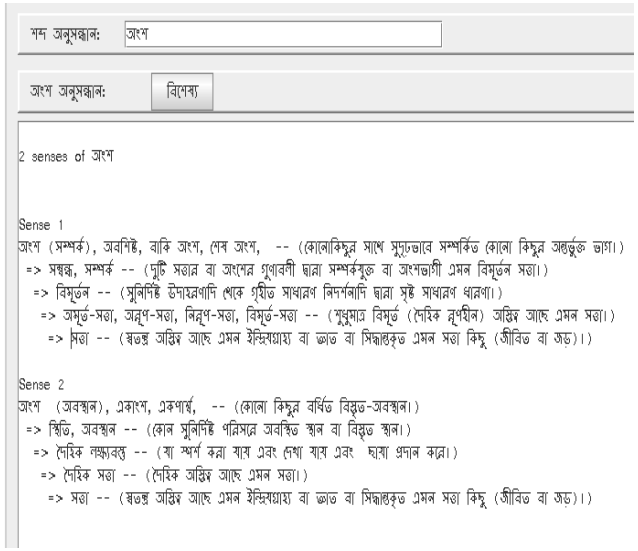


Diagram : A search result with hypernyms

Hypernym relation of a noun

3) Editing WNDB:

BWN supports editing any existing record through a user interface shown below. It also supports a limited version of delete operation, because an unrestricted deleted record may destroy the underlying tree. If the user wants to delete a record, there are three cases to consider:

- If the record has a synonym, then we can delete it (updates only the word table);
- If the record is used as a hypernym entry then we cannot delete it without risking relational integrity;

- If the record is not used as a hypernym entry, then we can delete that record, which affects all tables except the tree table. (Faruqe and Khan, 2008)

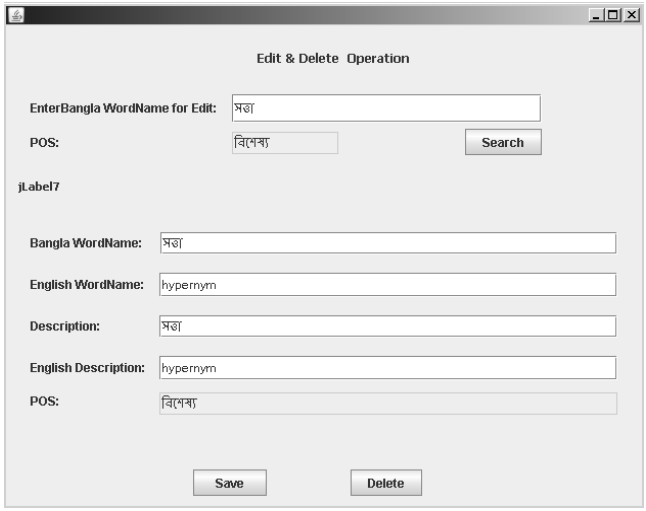


Diagram : BWN edit interface

Edit interface

BWN at the basic level supports building the WordNet database from lexical source files using a grinder, and then supports querying the data using an interface; in addition, it has two key features not found in other designs support for incremental building of the WordNet database, and for editing the WordNet data using an interface. These two key features significantly contribute to the complexity of the design and implementation of BWN. As BWN makes no assumption about the underlying language, it should be extendable to other languages as well. (Faruqe and Khan, 2008)

4. Part of Speech Tagging

4.1 Tagset

Despite several sparse attempts at automatic POS tagging of Bangla there has been a lack of a standardised comprehensive tagset. Hence the primary problems regarding POS tagging appear to be twofold: the absence of a comprehensive balanced corpus and the lack of a

standard tagset. Despite the dearth of a comprehensive annotated corpus, some recent evidence of experimental POS tagging using stochastic models can be found using different tagsets. (Dandapat et al. 2004; Dandapat et al. 2007; Ekbal et al. 2008). The results are indicative that the accuracy of the POS tagger can be significantly improved through by integrating a morphological analyzer, affixation information, name entity recognizer etc. (Mahmud and Khan, 2009). Since POS tagging can provide a stepping stone towards a building a syntactically annotated corpus it would be practical for the annotator to have a detailed POS tagging guideline. With this focus in mind CRBLP has created a POS tagging guideline for annotating Bangla to form a syntactic tree bank. The resultant tagset thus designed is based on the SPSAL tagset (SPSAL 2007) which has been proposed as a common framework for all Indian languages and analyses for disambiguating the confusing examples have primarily been influenced by the Penn tagset for English (Santorini 1990). The CRBLP tagset is a 'flat tagset' in that it lists the categories applicable to a particular language. The corpus used for this project is the Prothom Alo corpus. During this project 30,000 words were manually tagged, amongst which 25,000 (appx.) had been used as the training set and the remaining 5000 were considered as the test set (Mahmud and Khan, 2009).

As syntactical bracketing is a task of shallow processing and the size of the tagset is an important factor, only postpositions, accusative and possessive case markers on nouns have been incorporated in this tagset. A separate category 'Suffixes' has been included to reflect these characteristics of morphology. When a noun or pronoun is takes on a suffix, the base form and inflections are separated by a plus sign (+). Verbs are categorized according to their form such as finite, non-finite etc. A summary of the tagset is given below. Although, syntactic distribution has been considered as the main criteria while designing this tagset, there is also a conscious effort concerning encoding precise syntactic information. For example, the word 'কেন' (why) has been tagged as a separate category Question Adverb (QRB), rather than being included into Question Word (QW) in accordance with SPSAL tagset (SPSAL 2007). The sub-categorizations has been done due to the fact that 'কেন' (why) can also act as a relativizer and simply tagged as QW, it can't be shown that that relative phrase is a 'Relative Adverbial Phrase' (QADVP). Thus, resulting in loss of useful syntactic information

in case of syntactically formalizing a relatively free word ordered language (Mahmud and Khan, 2009).

4.2 Taggers

No specific tagger has been developed specifically for Bengali however some widely used taggers have been used and experimented with. Previously a rule based POS tagger has been used however only rules for nouns and adjectives have been made explicit. (Hasan, UzZaman and Khan, 2007). Noteworthy work on POS tagging has been reported for Indian Bengali where an HMM based approach is used for tagging. (Hasan, UzZaman and Khan, 2007). Efforts of a suffix based tree tagger can also be found for Bengali (Hasan, UzZaman and Khan, 2007). as well as a hybrid POS tagger based on HMMs (Hasan, UzZaman and Khan, 2007).

A small comparative experimentation was conducted on three South Asian languages i.e. Hindi, Telegu and Bengali using stochastic taggers in order to test which one performs best. The experiments were conducted using n-gram based unigram and bigram, HMM based and Brill's transformation based tagger. Although limited the results of the experiments show that the Brill's transformation based tagger's performance is superior to the other approaches in all the experiments.

The training, development and test data provided for the SPSAL contest (Workshop on Shallow Parsing in South Asian Languages (SPSAL), 2007) was used for the experiment. Training data sets for each of the languages Bangla, Hindi and Telegu were used separately, to create the training corpora. The test data provided there was the testing corpora. All the data provided for the SPSAL contest uses the SSF format (Bharati, Sangal and Sharma, 2006), however for convenience all the data from the SSF format was converted to the much simpler format used by the Brown corpus, included in NLTK (Bird and Loper, 2006). The results of the experiments are given below.

The experiment was run with Unigram, Bigram, HMM and Brill's tagger on Bangla, Hindi and Telegu. For Bangla a training corpus with a maximum of 1786 sentences consisting of 25426 tokens was used. The test corpus consisted of 400 sentences and 5225 tokens. The performances of the taggers are shown below (Hasan, UzZaman and Khan, 2007 : 4):

Table 1. Performance of taggers on the Bangla corpus.

HMM	Unigram	Bigram	Brill
63.6	56.9	55.5	69.6

Soruce: Hasan, UzZaman and Khan, 2007 : 5.

Table 2. Performance of taggers on the Hindi corpus.

HMM	Unigram	Bigram	Brill
68.5	58.5	57.5	71.5

Soruce: Hasan, UzZaman and Khan, 2007 : 5.

Table 3. Performance of taggers on the Telegu corpus.

HMM	Unigram	Bigram	Brill
56.6	42.8	42.2	66.9

Soruce: Hasan, UzZaman and Khan, 2007 : 5.

Brill's tagger achieves accuracies of 69.6% using 25426 tokens for Bangla, 71.5% using 26148 tokens on Hindi and 66.9% using 27511 tokens on Telegu, whereas the HMM tagger manages to obtain 63.6%, 68.5% and 56.6%. The Unigram and Bigram taggers manage 56.9%, 58.5% and 42.8%; and 55.5%, 57.5% and 42.2% respectively, using the same number of tokens as Brill's tagger. These results are also comparable and fall in the same range as those of the SPSAL contest (Hasan, UzZaman and Khan, 2007).

Based on these results the CRBLP tagset was tested using Brill's transformation based tagger.

4.3 Tagging issues and future work

The scope of the POS tagging guideline as outlined by CRBLP can productively be extended with many more examples and corresponding linguistic analysis for disambiguation. In fact the tagset itself is under a development phase. In 'Fineness vs. Coarseness' issue the tags are always chosen to be kept 'coarse' (Bharati et al. 2006; SPSAL 2007), because as number of tags are decreases the accuracy of manual tagging increases. Since the tagset presented is much finer in distinction, it is easy to see that certain modifications to this design are required. Keeping in mind the enhancement of both coarseness and linguistic aspect, it has been decided

that the tagset should be modified further according to following three issues at the current point of this stage:

1. Locative Nouns (NNL) and Temporal Nouns (NNT) have similar syntactic distribution. Thus they can be subsumed under a single category which is Spatial Noun (NST) as suggested in AnnCorra (Bharati et al. 2006).
2. Intensifiers such as খুব, বেশি, অনেক, have been categorized as adverbs. But it has been observed that their syntactic distribution is not as same as adverbs, because intensifiers always tend to appear before adjectives or adverbs that they intensify, where adverbs are allowed to appear anywhere in a sentence. Thus, intensifiers should have different categorization (INTF) (Bharati et al. 2006; SPSAL 2007).
3. Reduplication in order to indicate emphasis, deriving a category from another category. For example আস্তে আস্তে (slowly slowly), ছোট ছোট (small124.small), লাল লাল (red red) have been tagged using the same tag for both words such as আস্তে/RB আস্তে/RB. But reduplication is a highly productive process and it has been proposed in AnnCorra (Bharati et al. 2006) to include a new tag RDP for annotating reduplicatives. The current assessment decides to adapt this technique where the first word will be tagged by its respective syntactic category and the second word will be tagged as RDP, as for example আস্তে/RB আস্তে/RDP, to indicate that it is a case of reduplication distinguishing it from a normal sequence (Mahmud and Khan, 2009).

The tagging guideline and tagset thus outlined serves to create an opportunity for further scrutinization and investigation to build more comprehensive and explicit tagsets.

6. Conclusion

As the world gravitates towards a digitally-literate global society, Bangla computing has become integral in the evolutionary path for the language. Despite being one of the most widely spoken languages of the world, Bangla is one of the most digitally under-resourced languages. Various efforts for computational modeling can be noted as setting the precursors for a robust repository of computational resources for Bangla. Although the current description is not

exhaustive, it presents a detailed description of projects undertaken. The research conducted so far sets the stage for further work, which can be compounded on, and pave the way for building an extensive digitised repository of computational tools for Bangla.

References

A Part of Speech Tagger for Indian Languages (POS Tagger). 2007. Workshop on Shallow Parsing in South Asian Languages (SPSAL), Twentieth International Joint Conference on Artificial Intelligence.

Akshar Bharati, Rejeev Sangal and Dipti M Sharma. 2006. Shakti Analyser: SSF Representation. IIT, Hyderabad, India.

Altaf Mahmud and Mumit Khan. 2009. Unpublished Technical Report. CRBLP, Dhaka, Bangladesh.

Ayesha Binte Mosaddeque, Naushad Uz Zaman and Mumit Khan. 2006. Rule based Automated Pronunciation Generator, Proc. of 9th International Conference on Computer and Information Technology (ICCIT 2006), Dhaka, Bangladesh.

Barbu, Eduard and Verginica Barbu Mititelu. 2005. Automatic Building of Wordnets. Proc. International Conference Recent Advances in Natural Language Processing, Borovets, Bulgaria, pp. 329-332.

Bharati, A., Sharma, D. M., Bai, L. and Sangal, R. AnnCorra. 2006. Annotation Corpora for POS and Chunk Annotation for Indian Languages. Language Technologies Research Centre, IIIT, Hyderabad, India.

Colin P. Masica. The Indo-Aryan Languages. 2001. Cambridge University Press. New York

Debasri Chakrabarti, Gajanan Rane and Pushpak Bhattacharyya. 2004. Creation of English and Hindi Verb Hierarchies and their Application to English Hindi MT. International Conference on Global Wordnet (GWC 04), Brno, Czeck Republic.

Dewan Shahriar Hossain Pavel, Asif Iqbal Sarkar and Mumit Khan. 2006. Collaborative Lexicon Development for Bangla, Proc. International Conference on Computer Processing of Bangla (ICCPB-2006), Dhaka, Bangladesh.

Dr. Md. Shahidullah. Bangla Bhashar Itibritta. 2002. Mowla Brothers. Dhaka

Fahim Muhammud Hasan, Naushad UzZaman and Mumit Khan. 2007. Comparison of Unigram, Bigram, HMM and Brill's POS Tagging Approsches for some South Asian Languages. Proc. Conference on Language Technology CLT07, Pakistan.

Faruqe, Farhana and Mumit Khan. 2008. BWN - A Software Platform for Developing Bengali WordNet. International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering (CISSE 08).

Farreres, Xavier, German Rigau and Horacio Rodriguez. 1998. Using WordNet for building WordNets. Proc. COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems, Montreal.

Firoj Alom. 2009. Unpublished Technical Report. CRBLP, Dhaka, Bangladesh.

Gordon, Raymond G., Jr. (ed.), 2005. Ethnologue: Languages of the World, Fifteenth edition. Dallas, Tex.: SIL International. Online version: http://www. ethnologue.com/. accessed on 2016

K.R. Beesley. 2001. Finite-State Morphological Analysis and Generation of Arabic at Xerox Research: Status and Plans in 2001. ACL Work-shop on Arabic Language Processing: Status and prospects (Invited talk).

Manish Sinha, Mahesh Reddy and Pushpak Bhattacharyya. 2006. An Approach towards Construction and Application of Multilingual Indo-WordNet, 3rd Global Wordnet Conference (GWC 06), Jeju Island, Korea.

Md. Zahurul Islam and Mumit Khan. 2006. JKimmo: A Multilingual Combinational Morphology Frame-work for PC-KIMMO. ICCIT, Dhaka, Bangladesh.

Microsoft: Developing OpenType Fonts for Bangali Script. November 2002. url: http://www.microsoft.com/typography/OpenType%20Dev/bengali/intro.mspx. accessed on 2016

P. Sengupta and B.B. Chaudhuri. 1996. Morphological processing of Indian languages for lexical interaction with application to spelling error correction, Sadhana, Vol. 21, Part. 3, pp. 363-380.

Barbara F. Grimes, Editor 1996. Ethnologue, 13th Edition, Summer Institute of Linguistics, Inc.

Piek Vossen. 1999. EuroWordNet: A Multilingual Database with Lexical Semantic Networks, Computational Linguistics, Volume 25, Number 4.

S. Bhattacharya, M. Choudhury, S. Sarkar and A. Basu. 2005. Inflectional Morphology Synthesis for Bengali Noun, Pro-noun and Verb Systems. Proc. of the National Conference on Computer Processing of Bangla, Dhaka, Bangladesh, March, pp. 34 - 43.

S. Dasgupta and M. Khan. 2004. Feature Unification for Morphological Parsing in Bangla. Proc. 7th International Conference on Computer and Information Technology (ICCIT 2004), Dhaka, Bangladesh.

S. Dasgupta and M. Khan. 2004. Morphological Parsing of Bangla Words Using PC-KIMMO. Proc. 7th International Conference on Computer and Information Technology (ICCIT 2004), Dhaka, Bangladesh.

S.K. Chatterjee, The Origin and Development of the Bengali Language.2002. Rupa and Co.

Steven Bird and Edward Loper. 2006. Natural Language Toolkit. http://nltk. sourceforge.net. accessed on 2016

wordnet sql builder. 2008 http://wnsqlbuilder.sourceforge.net/schema.html accessed on 2015

Workshop on Shallow Parsing in South Asian Languages (SPSAL). 2007. Twentieth International Joint Conferences on Artificial Intelligence. Hyderabad, India.

Yeasir Arafat, Md. Zahurul Islam and Mumit Khan. 2006. Analysis and Observations from a Bangla news corpus. Proc. of 9th International Conference on Computer and Information Technology (ICCIT 2006), Dhaka, Bangladesh.