

Some Corpus Access Tools for Bangla Corpus

NILADRI SEKHAR DASH
Indian Statistical Institute, Kolkata

ABSTRACT

The techniques and strategies that are used to develop some Corpus Access Tools (CATs) for Bangla, as a part of Bangla Language Tool Kit (BLTK) are reported here. These tools are useful for retrieving linguistic data and relevant information from the modern Bangla written text corpus. At the initial stage only three tools are developed – Word Search Tool (WST), Collocation Search Tool (CST), and Sentence Search Tool (SST), which are combined into a single graphical user interface so that it can work quite elegantly to retrieve required data and information from the corpus. The hurdles that are encountered while trying to develop these tools are also addressed in this paper. Moreover, the problems and the solutions to these problems are also explicated here so that future researchers do not face much trouble to generate xml files to design new types of corpus access tool. One can visualize direct application of these tools in lexical search, concordance, collocation, language teaching, dictionary compilation, and language description. The strategies and techniques that have been adopted for developing these tools for the Bangla language corpus may also be utilized successfully to create xml files and similar tools for corpus processing for other Indian languages.

Keywords: corpus, Bangla, word search, collocation, sentence, part-of-speech, tagging

1. INTRODUCTION

The idea of developing Corpus Access Tools (CAT) was conceived when some Bangla corpus users asked for such

devices to retrieve relevant linguistic information, data and examples from the Unicode-compatible TDIL (Technology Development for Indian Languages) Bangla written prose corpus, which is now available in Internet for free download. To serve their requirements, we have developed some simple corpus access tools based on which the end-users are able to perform simple search functions like word search, collocation search, sentence search, etc. quite easily and quickly on the Bangla corpus. Based on the arguments of earlier scholar (Biber 1993), we believe that this tool will enable language users to retrieve necessary linguistic data, information and examples from the Bangla text corpus to address their language-related queries and questions.

The entire process of tools development is divided into two stages: (a) converting UTF-8 encoded Bangla text files (i.e., corpus files) into xml files by adding xml tags to sentences, words, numerals, alphabets, and other symbols (Bird & Loper 2004), and (b) making xml files compatible for word search, collocation search, and sentence search (Bird 2005; Bird 2006). The advantages of xml files and corpus access tools are that these are useful for the people working in linguistic query answering, natural language processing, language technology, language teaching, word search, dictionary compilation, and language description in Bangla.

In Section 2, I refer to some of the early works done in this area to show how it is characteristically different from the works claimed to be done by others; in Section 3, I propose to employ a set of principles for developing these tools for Bangla corpus; in Section 4, I refer to the basic functions of the Bangla corpus search tools with reference to the Bangla corpus on which the tools run; in Section 5, I describe the process of creating the customized xml files from the UTF-8 encoded text files automatically and then using these xml.dom-based search mechanisms for parsing the xml files to generate accurate outputs; in Section 6, I register the importance of such tools in various domains of natural language processing, language technology, language teaching, and language description. In

conclusion, I focus on the present state of such tools in Bangla as well as in other Indian languages corpora.

2. EARLY WORKS

There are some tools that are made to access corpus not only of English but for many other languages. These tools, in actual sense, differ in goal and application from the tools reported in this paper. An example of this is the *WordSmith* (<http://www.lexically.net/wordsmith/>) tool that is used quite exhaustively for many languages across the world. This particular tool, however, does not support the Bangla language corpus, although it claims it supports to some of the Indian languages like Urdu and Hindi.

The next one is *Google n-gram viewer* where one can search not only for word co-occurrence statistics but also how the patterns change over a long period of time (<https://books.google.com/ngrams>). This tool mostly operates on digital corpora procured by Google relating to languages like American English, British English, Chinese, German, Hebrew, Italian, Russian, Spanish, Dutch, French, etc. Moreover, since it allows parts-of-speech-based search, it requires corpora tagged at the part-of-speech level. This tool can be an excellent utility for Bangla and other Indian languages, but unfortunately there is no Indian language diachronic text corpus, which is available in POS tagged version to be uploaded and accessed by this tool. Until and unless Google uploads similar types of text corpora of Indian languages into its server-based interface, this will not come to help to the Indian languages. Although Indian languages are right now not supported by Google n-gram viewer, it inspires us to think for developing a tool of this kind which may be useful to have an option to display co-occurrences from Web, for which we can use n-gram options for Bangla text as well. Also it is possible to use Wikipedia dump for Indian languages to support similar queries.

The third effort in this direction is initiated by the Leipzig University, Germany through development of web-based interface for 230 Corpus-Based Monolingual Dictionaries

(<http://corpora.uni-leipzig.de>). Although the developers claimed that they have incorporated corpora of several Indian languages like Goan Konkani, Gujarati, Malayalam, Panjabi, Tamil, Telugu, Bangla, Bishnupriya, Fiji Hindi, Kannada and Western Panjabi, etc. into their collection, in reality, it is not clear how these corpora are procured, processed, and stored into the lexical database used for the monolingual dictionaries. While we searched their corpus-based monolingual dictionaries, we came out with no result – every time we are asked to try later as the database is under maintenance. Although it seemed to be a highly useful tool for lexicographic search for the text of Indian languages, the system is not yet built up to support lexical-based search in corpora – as our tool is able to do.

The fourth and more recent attempt is made by Society for Natural Language Technology Research, Kolkata (<http://nltr.org>) who digitize a large volume of Bangla literary texts, which, as claimed, are searchable through a simple web-based interface that works on-line. Although this searching interface appears to be simple in operation it is not platform-independent and it is not useful for object-oriented linguistic search where one is interested to identify different lexical items in the corpus with specific pre-defined contextual frames – a highly sophisticate and intelligent interactive interface, which has been adequately addressed in the tool developed by us. We are seriously thinking for integrating these literary texts developed by Society for Natural Language Technology Research into our interface as well. However, it is not possible as these resources are in highly restricted use (available only for on-line reading) without any scope for using the texts as training corpus database.

3. PRINCIPLES ADOPTED FOR CAT

Following the observations of earlier scholars (Shannon 2003), it was clearly realised that unless the basic principles behind development of tools are defined it may create problem in justification of the tools in works of corpus processing. Therefore, some principles are identified and laid down in clear terms to make the entire process of tool development systematic,

coordinated and unambiguous. Moreover, these principles are adopted and followed rigorously in every stage of tool development to give robust applicationality of the tools on the Bangla corpus. The principles that are adopted and followed are as follows:

Principle 1: XML encoding for Sstorage

The raw text files of .txt format are not very suitable for storage in a server, because processing of such files is time consuming and inefficient. So these files are required to be converted into a format which is more server-friendly. Hence the xml format is opted for the better storage of the text files. This helps the tool developers (a) to define their customised tagset by which they can tag the text data easily as well as all mark the features of words in accordance with the requirements of a particular tool, and (b) at later stages, they can easily fetch target data based on the tagset and features assigned to the texts (Klein 2006). Besides, the xml format is extensively used to store the metadata associated with each text file. This gives an option for using either raw xml files or storing the xml files within a database and using it. The former option is preferred to other for obvious reasons. The main reasons behind selecting xml files for data storage are the followings:

1. There are many utility libraries and APIs for xml files. These make the process of extracting tag contents or selecting portions of a text or searching document headers much simplified.
2. Future enhancement of properties in text is easier in xml format. In regular format the database needs to be redesigned for increasing enhancements in search criteria. But in case of xml format addition of a few tags will do the trick.
3. Effective word tagging and retrieval technique based on xml format is much effective in text database stored in the system.
4. As data security is not a big issue in this case, there is no need to burden the corpus database with additional intratextual annotation in xml files.

5. Future integration of xml files with additional text databases is easy and trouble free.

Thus it is pertinent to argue that the use xml format is crucial as it does not require the use of any particular software. It is highly flexible and simple in text database management. Besides, it builds up a huge support baseline into standard utilities like web browsers, database access systems, easy retrieval, and data display, etc.

Principle 2: Uniformity

The techniques that we had used to create the tools are not specifically designed to be useful for the Bangla text corpus only. It can also be used for corpus of other Indian languages including English. The tool will function quite effectively in those cases as well.

Principle 3: Universality

The techniques we had used to build the xml tagged sentences from the corpus are at par with the technology used for other languages, say, English, French or Spanish. Following the file formats used to store the *British National Corpus* and the *Australian Corpus of English* we had used the xml format as our chosen file and storage format.

Principle 4: Extensibility

The tagset that we had used is open for extension and modification of any kind. Moreover, if it is required, it is also possible to do further filtering of data and add more metadata into the xml files. That means by adding more tags to the sentences we can refine the search queries based on more fine-tuned criteria.

Principle 5: Expandability

The module allows us to expand the load of data in the corpus, if required, at any point of time. Moreover, the new text files that are added to the existing corpus database can also be converted into xml. These xml files can be uploaded to the server and be

made ready for use within a very short span of time with little effort.

Principle 6: Dispensability

If required, all the tagged xml files can easily be converted into untagged xml files within a few steps without invocation of any complex process of text conversion. That means the original text database can be easily retrieved from the corpus tagged with metadata.

Principle 7: User friendly

The meaning and the purpose of the tagset used for our work are kept maximally open so that future enhancement of the existing coding system or the method is easily carried out.

Principle 8: One-to-one output

In our module, a sentence has just one xml tagged form. It does not allow a sentence to have two different xml tagged forms. In the reverse scheme – reverting from an xml tagged sentence to a normal one will uniquely give the sentence from which it is generated. This implies that no ambiguity is permitted to arise within the process of file conversion.

4. THE CORPUS ACCESS TOOL

The CAT is a java-based application, which is designed to perform word – collocation – and sentence-based search in the TDIL Bangla prose text corpus. In this context, it is necessary to refer to content and composition of the TDIL Bangla corpus on which our tool runs. We have downloaded this Bangla corpus from the web as it is made available as an open resource for research and development purposes. This corpus contains five million words of modern Bangla prose texts that were published between 1981 and 1995. The texts covered more than 85 subject domains with proportional representation of text samples from each domain to make the corpus maximally authentic about the present status of the language used in the state of West Bengal, India.

Initially the corpus was developed in the ISCII (Indian Standard Code for Information Interchange) format, and because of this format the corpus was not easy to use in any work of linguistics or language technology. In recent times, however, it is converted into Unicode compatible format and due to this fact the corpus is made available for global access and utilization. Moreover, the texts in the Unicode version are roughly normalized with removal of crude orthographic errors and junks. At present the corpus has 1270 files – each file containing nearly 500 sentences in the form of a continuous text without any pause/break at the end of a sentence or a paragraph (Dash 2007).

While formatting the files we observed that each file had two major parts. The first part contained the elaborate metadata in the form of a Header File that recorded the details of extra-textual information relating to the source of the text. The second part, on the other hand, contained the Bangla prose text in standard Bangla orthography.

We had to use this particular corpus as our primary text database because there was no other corpus so well framed and well formatted to be maximally useful for our works. Moreover, this corpus had also been the primary database for verification and validation of the search functions that are currently supported by the CAT we have developed. At present the CAT supports the following three search functions (Christ 1994):

1. Word Search,
2. Sentence Search, and
3. Collocation Search

Each search function is explicated with some examples obtained from the corpus under our disposal.

4.1. *Word search*

The word search tool works for finding out that particular word, which a user wants to find out in the corpus. At first a user enters a Bangla word (or a substring of it) into the search interface. Since the pipeline does not include the process of lemmatization, the sub-string search algorithm does not yield information about

lemmas but matches the input sub-string with that of a larger string to catch a word. The tool then takes the word as an input, searches through the corpus, and shows up all the sentences that contain that particular word (or the substring of it) as these are found in the corpus. It also provides user with the sentence number and the file names which contain the word that the user searches. The user has the liberty to specify a particular filename or a topic name, if the user wants, to restrict the search to a particular file or a specific domain or a subject topic.

This strategy has several functional advantages. First, it not only searches out the target word a user wants from the corpus, but also helps a user to find out different forms of a particular word (e.g., affixed, inflected, non-inflected, duplicated, compounded, etc.) used in the corpus, along with the details of their usage patterns, frequency of use in a particular topic, and different meanings the words evoke in various contextual frames they occur. A simple search for the word মাথা [*māthā*] “head”, for instance, produces more than hundred different sentences from the corpus, based on which it is possible to sub-classify manually the sentences (S) further in accordance with their meanings (M) denoted by the word in its occurrence within different syntactic-cum-semantic contexts (C), as the following examples and the diagram (Figure 1) show.

- a. সে এই দলের মাথা
se ei daler māthā
“He is the leader of this gang”
- b. তার মাথা ফেটেছে।
tār māthā pheṭeche
“His head is bleeding”
- c. পাহাড়ের মাথায় তার বাড়ি।
pāhāṛer māthāy tār bāḍi
“His house is at the top of the hill”
- d. এবার মাথায় চিরুনি দাও।
ebār māthāy chiruni dāo
“Now, comb your hair”
- e. আঙ্গুলের মাথাটা ব্যথা করছে।
āṅguler māthāṭa bythā karche
“The tip of the finger is paining”

- f. চার মাথার মোড়ে তার দোকান।
chār māthār moṛe tār dokān
 “His shop is at the crossing of four roads”
- g. ছেলের অঙ্কে বেশ মাথা।
cheleṛ aṅke beś māthā
 “The boy is intelligent in Mathematics”

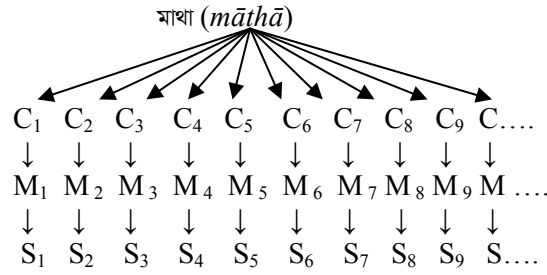


Figure 1. *Sub-classification of sentences based on meaning*

This work, however, is done manually, since S, C, and M level information is NOT coded into the corpus and the search does not yield this output. The search is only giving out a syntactic frame from where S, C, and M information can be manually retrieved.

Since the tool also provides an option for substring- based search, a user can find out different structural forms of the same word from the corpus. For example, substring-based search for the word কর (*kar*) extracts several forms of different grammatical categories to show how various surface forms of different parts-of-speech, which exhibit orthographic proximities, may be clustered together under one base/root (Figure 2).

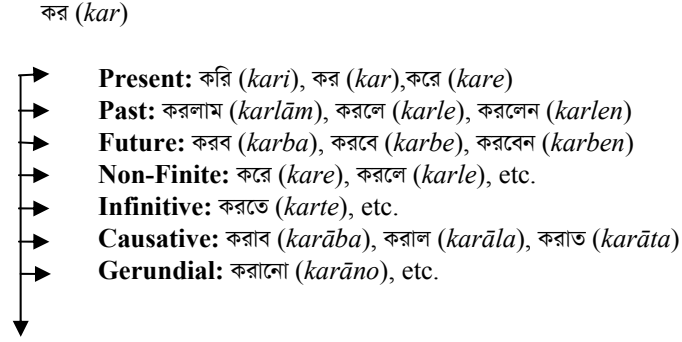


Figure 2. *Clustering of words under single base/root*

This, however, also yields a lot of ambiguous results because the substring-based search for the word কর (*kar*) also extracts many other forms that do not belong to this set. Rather than considering it as a fault of the tool, we consider it as an advantage, as it produces all the words that are made up with the sub-string কর (*kar*) at initial position. Further scheme of lexical classification will ensure that words are classified in appropriate lexical groups even though these are initially compiled together as an obvious consequence of a mere orthographic match.

This allows a user not only to assemble simple forms of the base/root but also helps him/her to assemble all the inflected forms (and other forms) of the word available in the corpus, as the above diagram shows. After studying the association patterns of other words with কর (*kar*) or all other forms generated from কর (*kar*), a user easily traces conceptual proximity or semantic closeness among the forms with regard to their usages in the natural Bangla texts (Bird, Klein, Loper & Baldridge 2008). Based on this kind of observation we may formulate a hypothesis that words representing similar orthographic structures may have some conceptual alliances and semantic affinities. It is also possible to infer that they are derived from the same etymological antiquity due to which they exhibit almost similar semantico-grammatical patterns of occurrence in the language (Holmes, Ahmad & Abidi 1994).

4.2. Collocation search

The next tool that we have developed can list up and show up all collocated words based on user input. First it invites a user to enter a Bangla word (Target Word = TW) in the interface. Once the TW is entered, it tries to retrieve all the words that are used immediately before (Type-I) and after (Type II) the TW in the corpus. Then it lists up all the word pairs and displays the same in a frequency-based sequential order in which the most frequently used word-pair occupies the first position followed by others, as the examples show (Table 1).

Table 1. Collocation pattern of হাত (*hāt*) “hand”

Type-I:		
মাথায়হাত	<i>māthāy hāt</i>	“hand on head”
বুকেহাত	<i>buke hāt</i>	“hand on chest”
মুখেহাত	<i>mukhe hāt</i>	“hand on mouth”
গায়েহাত	<i>gāye hāt</i>	“to punish”
পায়েহাত	<i>pāye hāt</i>	“to show respect”
মুখহাত	<i>mukh hāt</i>	“face and hand”
কালোহাত	<i>kālo hāt</i>	“black hand”
জলহাত	<i>jal hāt</i>	“wet hand”
একহাত	<i>ek hāt</i>	“one hand”
আপনাহাত	<i>āpnā hāt</i>	“self hand”
Type-II:		
হাতপা	<i>hāt pā</i>	“hand and leg”
হাতমুখ	<i>hāt much</i>	“hand and face”
হাতসাফাই	<i>hāt sāphāi</i>	“stealing”
হাতকাটা	<i>hāt kāṭā</i>	“cut off hand”
হাতকড়া	<i>hāt kaṛā</i>	“handcuff”
হাতছানি	<i>Hātcāni</i>	“hand waving”
হাতপাখা	<i>hāt pākhā</i>	“hand fan”
হাতপাতা	<i>hāt pātā</i>	“begging”
হাতটান	<i>hāt ṭān</i>	“miserliness”

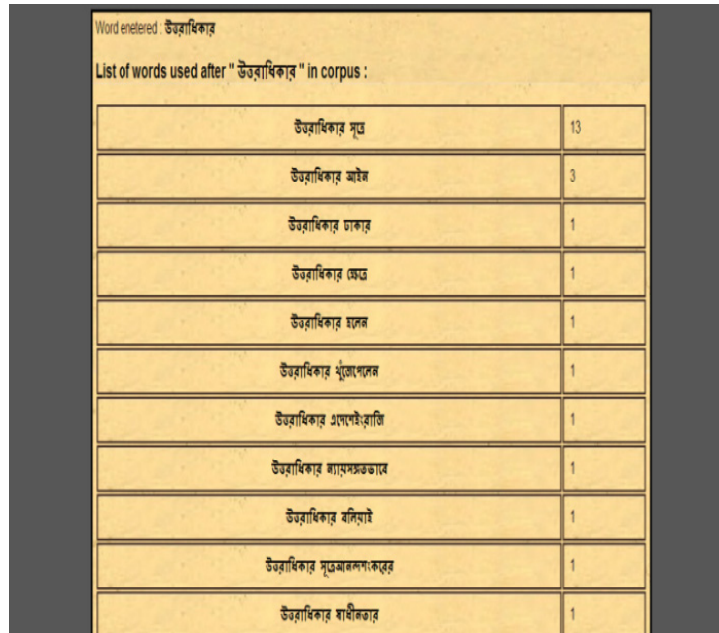
To explicate the process of collocation search we have presented below a list of collocations made with the word মাথা (*māthā*) (these are fished out from the corpus with our word-search tool) to show how the word is able to exhibit its usage patterns and meaning variations within different collocation frames obtained from larger syntactic strings (Table 2).

Table 2. Search results corresponding to মাথা (māthā)

মাছেরমাথা (mācher māthā)	“head of fish”
ছাতারমাথা (chātār māthā)	“useless thing”
আলমারিরমাথা (ālmāirmāthā)	“top of cupboard”
গ্রামেরমাথা (grāmer māthā)	“village head”
পাহাড়ের (pāhāḍer māthā)	“peak of a hill”
আঙ্গুলেরমাথা (āṅguler māthā)	“finger tip”
জলেরমাথা (jaler māthā)	“water surface”
নদীরমাথা (nadir māthā)	“source of river”
রাস্তারমাথা (rāstār māthā)	“end of road”
রাস্তারমাথা (rāstār māthā)	“crossing of roads”
পাকামাথা (pākā māthā)	“sharp brain”
ক্ষুরেরমাথা (kṣurer māthā)	“blade of razor”
কাজেরমাথা (kājer māthā)	“sense of work”
অঙ্কেরমাথা (aṅker māthā)	“knowledge of Maths”
দলেরমাথা (daler māthā)	“captain of team”
গ্যাঙেরমাথা (gyāṅer māthā)	“gang leader”
নৌকারমাথা (naukāṛ māthā)	“head of boat”
অফিসেরমাথা (aphiser māthā)	“boss in office”
কোম্পানিরমাথা (compānir māthā)	“company owner”
পরিবারেরমাথা (paribārer māthā)	“head of a family”
মাথায়মাথায় (māthāy māthāy)	“of equal height”
কাগজেরমাথা (kāgajer māthā)	“margin of paper”
বিছানারমাথা (bichānār māthā)	“edge of bed”
দেশেরমাথা (deser māthā)	“president of country”
ফোড়ারমাথা (phoḍār māthā)	“mouth of pimple”
মুদ্রারমাথা (mudrār māthā)	“side of a coin”
পেনেরমাথা (pener māthā)	“cap of pen”
সোজামাথা (sojā māthā)	“straight face”
উঁচুমাথা (ūchu māthā)	“high pride”

In reality the tool categorically identifies the number of times each TW is noticed to be used in the corpus in its specific collocational environments. Here again a user can also monitor and restrict his/her search to a particular topic or a particular file by way of specifying the filename or the topic name of a text. Although we know that the efficiency of a tool is substantially improved by adding techniques for noise removal; replacement

of raw and noisy corpus with a normalized and noise-free one; and replacement of un-annotated corpus with a POS tagged corpus, we have not been able to apply our tool on a noise-free POS tagged corpus to increase the level of its accuracy because such a resource is not available in Bangla. Since this tool does NOT operate on POS based information to retrieve results, the results are not compared to any generic collocation search. We have given before a screen shot which displays some examples of collocation of a word derived from the corpus (Figure 3).



Word entered: উত্তরাধিকার	
List of words used after "উত্তরাধিকার" in corpus :	
উত্তরাধিকার সূত্র	13
উত্তরাধিকার আইন	3
উত্তরাধিকার চাকার	1
উত্তরাধিকার ক্ষেত্র	1
উত্তরাধিকার হালনা	1
উত্তরাধিকার খুজাপেলন	1
উত্তরাধিকার এদেশেই:রাতি	1
উত্তরাধিকার ল্যাংসঙ্গভারে	1
উত্তরাধিকার বলিয়াই	1
উত্তরাধিকার সূত্রানলশাকার	1
উত্তরাধিকার স্বাধীনতার	1

Figure 3. Collocation of উত্তরাধিকার (uttarādhikār) “inheritance”

4.3. Sentence search

The third tool that we have developed can be used on the corpus to generate a list of sentences that contain specific numbers of words. A user has to provide the exact number of words supposed to be present in a sentence and the tool will fish out from the corpus all the sentences that are made of with that specified number of words along with respective filenames and

sentence numbers. Here again the user is free to restrict his/her search to a particular topic or a file by way of specifying the filename or the topic name (Vaughan & Thelwall 2004).

This tool is highly useful for finding out the particular type of sentences, which tend to be of smaller or larger in size with regard to the number of words. For instance, if a user wants to find out sentences made with just two words, s/he has to specify the number of words in a sentence, and the tool will retrieve all the sentences made with just two words from the corpus as the following examples show.

- | | | | |
|----|------------|-----------------------|------------------------|
| a. | কোথায়চলে? | <i>kothāy challe?</i> | “Where are you going?” |
| b. | হেভাগবান! | <i>he bhagabān!</i> | “Oh my God!” |
| c. | বসুনআপনি। | <i>basun āpni</i> | “you sit” (Hon.) |
| d. | লিখেফেল! | <i>likhe phela!</i> | “Write down!” |
| e. | আপনিমহান। | <i>āpni mahān</i> | “you are great.” |

What is most striking in this application is that it fishes out not only the simple declarative sentences, but also sentences of other types including that of imperative, interrogative, and exclamatory ones. Such sentences are hardly used in statement or narration but used mostly in conversation and dialogic interaction. Since there is no coding for sentences in the corpus, it has become an easy task for the tool to count the number of words and get them listed in the final output.

The advantage of this application lies in its ability for extracting sentences based on the number of words from the specified files, topics or the whole corpus as well as its ability in capturing the finer discourse-related information embedded in the sentences. Thus, with the help of this tool a user can easily find out the sentences made up with varied number of words as well as can calculate their frequency of occurrence in different text types included in the corpus (Robinson, Aumann & Bird 2007). However, it should be clearly stated here that it is not an attempt to link sentence length with complexity. We have NOT tried this because we know that to establish such a link, we need to devise a matrix that will provide some parameters for sentence complexity. The frequency count that we get at word level for a sentence is a

very simple count without scaling the text types in the corpus (i.e., sports, finance, science, law, etc.). Given below is a screen shot that cites a list of sentences containing 15 words and these sentences are obtained from the Bangla corpus (Figure 4).

Sentences with 15 words are :

1	BAC08	000019	এই সিদ্ধান্তটি পূর্ণ: পূর্ণ: প্রকাশ করে যেনো যান যে, অতীশিয়া (desired) ফল পাওয়া যায় কিনা।
2	BAC08	000041	এই নব কারণে কোল নির্দিষ্ট বিষয়বস্তুর বিশ্লেষণ, মূল্যায়নবা ভবিষ্যৎপূর্বাভাস (prediction) করার জন্য ভূতের প্রয়োজন।
3	BAC08	000055	কারণ ঐ মূল্য একটাই এবং সেটা ঘটনার দ্বারা স্বীকৃত বলে প্রমাণ করার কোন অবকাশ থাকে না।
4	BAC08	000066	অখণ্ডতার নষ্টাব্দে দুই উচ্চ হলেও পণ্য বিক্রি না হলে তার ক্ষতি হলে পণ্য করানো হবে না।
5	BAC08	000079	কিছু কতটা তথ্য প্রকাশিত হলে সেটা যথেষ্ট নাহা ও সম্পূর্ণ হবে যে বিষয় তত্ত্বাবধায়ক।
6	BAC08	000148	সম্পত্তির ওপর ক্ষেত্রে বিশেষ কোনো বা মালিকের একমাত্র অধিকার আছে এবং সমস্ত ঐ অধিকার স্বীকার করে।
7	BAC08	000167	কারণ এর দ্বারা দুই খেটে মারতেও পণ্য গণনা ও তার মূল্য নির্ণয় করা সহজ হয়।
8	BAC09	000032	পরে ঐ ক্ষেত্রে সম্পত্তির কার্য ও তার ব্যয়ের মধ্যে তুলনা করে তার পার্থক্য নির্ণয় করা হয়।
9	BAC09	000043	কিছু বেশ প্রতিষ্ঠানের বিভিন্ন প্রকার মালিক দ্বারাও অন্যান্য দূর থেকে ধন করে আর্থ সংগ্রহ করা হয়।
10	BAC09	000052	ব্যবসার প্রয়োজনীয় সব টাকা (সেয়ারহোল্ডারের কাছ থেকে না নিয়ে কিছু অংশ ধন করে সংগ্রহ করা হয়।
11	BAC09	000092	চতুর্থত, ধন করা মূলধন ব্যবহার করে ইকুইটি (সেয়ারহোল্ডারের কাছ থেকে না নিয়ে কিছু অংশ ধন করে সংগ্রহ করা হয়।

Figure 4. Search of sentences containing 15 words

The main motive behind developing such tools is to provide linguistic researchers some useful devices for performing rudimentary tasks relating to search, count and sort out lexico-syntactic data and information available in the Bangla corpus (Sinclair 2004).

Although all the functions carried out by the tools can also be done manually, it will consume enormous amount of time to complete the tasks which, on the other hand, can be done by a computer within the shortest span of time. Moreover, in case of machine-based application, the probability of error is near to null, while in case of manual calculation, the amount of error is very high and not beyond question. Therefore, it is rational to use

automated search tools to increase the level of accuracy as well as efficiency in the tasks carried out on the corpus database. Furthermore, in the cases where trained human resource is not readily available, one can use these tools to execute research goals successfully. For instance, for collocation analysis of words, instead of spending the entire life of an investigator to find out words that are used immediately before and after the TW, a user can employ this tool to do the same task easily, quickly, and accurately. The accepted utility of the CAT is, in fact, equal to many Python and R libraries and packages as it allows users to accomplish a lot of what the tool promises to do and more.

5. IMPLEMENTATION

The entire scheme is envisioned to be implemented in four primary steps: (a) assessing user requirements; (b) creating xml files of the whole corpus; (c) creating user-friendly tools for handling xml files and generating the results for the end users; and (d) developing the user interface.

5.1. *Assessing user requirements*

Our primary goal is to develop tools for word as well as collocation search from the whole corpus database or from a particular file or a specific text type. Also, we wanted to develop a tool that counts number of words and sentences in a text file, in a topic or in the whole corpus. Moreover, we decided to develop a tool which will find out sentences containing specific number of words. In all these activities our main focus is to make the tools maximally responsive so that these tools are utilised by the end users without less complication (Arnold 2000). To achieve this mission, we are planning to collect user queries during beta test of the tool which can help us understand the needs of the users more accurately as well as redesign the tool (if needed) to gain from the generalizations drawn from the users' input. Moreover, since the tool is Java based, instead of XML, we are seriously thinking using JSON (JavaScript Object Notation) that

we hope may activate better interface for global access of the tools.

5.2. *Creating xml files of the corpus*

We are provided with UTF-8 encoded 1270 text files of the Bangla prose texts. Each text file contains metadata in the form of a Header File with 8 (eight) entries: file name, topic name, sub-topic name, year of publication, medium of publication, name of the published material in Bangla, author's name, and page number of the document from where the text is taken.

We have first developed a java program to create xml files from those text files based on the file names and/or topic names. At first we have identified 1270 files based on filenames. In the second stage, based on topic names, we have classified these files into nine (9) text categories: literature, fine arts, natural science, social science, commerce, school text, medical text, legal text, and mass media text. These major text categories cover all the files that we have used for our purpose.

Some text files, however, contain various sub-types of texts. To manage these files, we have classified them into 27 general xml classes, each one of which contain 50 files each irrespective of topic or filename. These are required because when a user tries to search into the total corpus database, these 27 xml classes will be parsed first. On the other hand, when a user will search in the files based on a topic name or a filename, the respective file of that name will be used to accelerate the search process.

We have thoughtfully adopted this strategy to avoid creating an exceptionally large xml file consisting of the whole corpus of 1270 text files. Hence the corpus is sub-divided into 50 files each and each file is converted into an xml file. Since the same text file can be used for different corpus-based searches (e.g., topic-based search, file-based search, etc.), we want to keep the backup files ready so that if one file-search algorithm fails (say, the topic-based search), a user who is searching for a word in the whole corpus, can still find the word with the help of other types of search.

The well-formed tagset is used to create the xml files because the corpus is the document root. It is divided into multiple files – each one having an attribute file number as well as a Header, a

Text, and an Extent as its children. The Header contains a single child type, which again contains a single child with specific child-type name. For instance, consider the field: Book. The subdivision of the type is done because if, in future, text data from other sources like newspapers, journals, pamphlets etc. are also added to the corpus, their Header field will vary to a large extent to register their separate subject category as well as linguistic identity. Moreover, by adding some extra field types a user can easily check which type of text data is obtained and from which source. The use of the Header format for this kind of operation, in our view, will be easier and effective instead of modifying the entire code set. For instance, the field “Book” contains the following six sub-heads:

(a) Header

- **Topic:** Topic of data, e.g., Commerce
- **Sub-topic:** Sub-topic of data, e.g., Accountancy
- **Author:** Name of author (if not specified, the field is left empty)
- **Time:** Time of publication (if not specified, it remains empty)
- **Book Name:** Name of the book in Bangla script, e.g., হিসাবশাস্ত্র (hisābśāstra) “Accountancy”.
- **Page:** Page numbers of the book from where the text data is taken.

(b) Text

Each text file contains large number of sentence(s) – each one having an attribute ‘sentence number’ (sen), children (sentence body) and the number of words in that sentence. The sentence body, in actuality, consists of words, alphabets, digits, punctuation marks, and other characters that constitute to form the structure of a sentence as a string of many words marked with empty spaces. The programmatic representation of the algorithm looks like the followings (Bloch 2005):

- **s:** sentence with sentence body and no. of words, have attribute n and children (sen) and “now.”
- **sen:** sentence body consisting words, punctuations, etc.

It is to be noted here that if part-of-speech tagging has to be done in future it can be added as an attribute to the field w.

- w: word
- p: punctuation having attribute type which can have values pnt, pnc, pul and pur.
 pnt: terminating punctuations which end a sentence like |, !, and ?
 pnc: punctuations where sentence doesn't end, like ; , : etc.
 pul: left brackets, [, {, (.
 pur: right brackets,], },).

The most notable thing is that both the left and the right brackets are marked because English meanings of some of the Bangla scientific words and technical terms are specified by enclosing them within brackets. If, for any reason, those meanings associated with words are required for some analysis, they can be used. On the other hand, if anyone wants to extract and display the Bangla words only, it can also be done by hiding the words that contain English meaning within brackets. Further, other symbols that are encoded to define specific textual functions in the texts are the followings:

- q : quotations “ , ”
- n: for numerals like 5 or 5th etc.
- a: for all single letter entities
- s: for symbols like &, @, #, \$, %, ^, *, <, >, -, _ , +, =, \ and /
- now: number of words in that sentence

(c) Extent

Additional information of words is embedded within the domain: Extent. That means it contains details about the extent of each file and has two children, as the followings:

- no_of_word: Total number of words in a file
- no_of_sentence: Total number of sentences in a file

Based on the strategies and techniques stated above each sentence in each text file of the Bangla text corpus is tagged in the following manner (Figure 5):

File Name: BACO8
Text Type: Commerce
Sentence No: 3
Untagged sentence: হিসাবতত্ত্ব সম্বন্ধে আলোচনা শুরু করার পূর্বে তত্ত্ব বলতে কি বোঝায় সেটা স্পষ্ট হওয়া দরকার।
Tagged sentence: <s n="000003"> <sen> <w>হিসাবতত্ত্ব</w><w>সম্বন্ধে</w><w>আলোচনা</w><w>শুরু</w><w>করার</w><w>পূর্বে</w><w>তত্ত্ব</w><w>বলতে</w><w>কি</w><w>বোঝায়</w><w>সেটা</w><w>স্পষ্ট</w><w>হওয়া</w><w>দরকার</w> <p type="pnt">।</p></sen> <now>13</now> </s>

Figure 5. Sample of a tagged Bangla sentence

At the end of this initial preparation phase we have the xml documents ready from where necessary data can be retrieved by the application of the tools.

5.3. Creating web-based tools

We have used JSP for creating the application, since it is platform-independent and very much convenient to work with. The code which we have used is very simple JAVA code for XML parsing that produces the result set, which is further refined to produce the final results that are expected from the programme. We have relied heavily on W3C document object model and the javax.xml parsers. In most cases, we have used xml.dom parsing for searching words, substrings of words, and collocations besides other functions (Brill 2002; Frath & Gledhill 2005).

The primary advantages of using JAVAX.XML and the W3C.DOM libraries are that these operations are less complex in nature and consume less amount of time to parse the xml files due to which retrieval of results often becomes much faster. Thus it reduces the load of latency of the application quite significantly.

The resultant output, at the end of this phase, is the development of a functional application based interface, which is highly capable of performing all the three major functions specified in Section 2.

5.4. *Developing the user interface*

The actual user interface is developed in a simple manner minimizing the amount of complexities in the operation of the system. Keeping the needs and levels of efficiency of the end-users in mind, the interface is developed in such a manner that it is able to alleviate the load of work of the end users as much as it can. A simple look at the interface will show that it is self-sufficient and self-explanatory (Figure 6).



Figure 6. *A simple interface of the BCST*

We believe that any one, following our strategy and techniques, can easily develop similar corpus searching tools for any Indian language corpus. There are, however, some pre-defined guidelines that cannot be ignored at the time of developing such an interface. The guidelines are the followings:

1. The text files of the corpus should have same header format with eight entries,
2. Each word, number, symbol, as well as every separate lexical entity should clearly be demarked with a space before and after it in the normalized corpus.

If these two basic guidelines are followed then the same JAVA program that we have used here can also be used for creating xml files from any kind of text corpus of Bangla as well as from other natural languages across the synchronic range and diachronic variations. Moreover, the xml files from corpora of other Indian languages can also be created with some rudimentary modifications of the program designed by us. Furthermore, once the xml files are created from JAVA program, the same set of JSP applications that we have designed can be used on other Bangla corpora or the corpora of other Indian languages. This observation stands valid for word-based and lexical collocation based search operations as well as for similar functions that are applied on digital text corpora.

6. IMPORTANCE OF THE XML FILES AND TOOLS

The importance of xml files of digital texts is immense in various domains of linguistic and language technology related research and application. These xml files, along with basic word tagging techniques, can be aptly utilized for part-of-speech tagging of words as well as for web-based application like searching, sorting, and analyzing POS tagged texts. In descriptive and applied linguistics such files are highly useful for frequency calculation of words, lexical sorting, basic vocabulary compilation, and language teaching (Anthony 2004).

On the other hand, the utility of corpus-accessing tools is enormous in natural language processing, language technology and empirical linguistic studies. We visualise that the CAT that we have developed, can be utilized by language researchers in various NLP applications, such as, developing interfaces for grammar checking, developing systems for capturing usage patterns of words, making concordance of word use, grouping

local words, defining distribution patterns of words, identifying patterns of lexical combination, lexical collocation, parsing sentence, addressing queries, modelling linguistic search in corpus, information retrieval, extraction of grammatical properties, text understanding, E-learning, E-governance, and many other works (Hearst 2005; Wallis and Nelson 2001).

7. CONCLUSION

Since there are several merits of having corpus-search tool, the present attempt should be appreciated in open heart although, on technical merits, it needs some amount of polishing to make it platform independent and user-friendly. To make it happen we are trying to arrange the output from the search queries in a more usable format, such as CSV (Comma-separated values) file format, so that future statistical language models could be trained on the query output. In fact, the basic summary statistics that are elicited from this format should also form a part of the query results referring to the overall distribution of the target lexical items in the corpus database vis-à-vis in the language. Moreover, to make this tool maximally useful, we need to apply some development iterations before it could become truly indispensable in the area of corpus-based lexical search.

Although we can visualize many more applications of these tools in Indian language corpora, not much effort has been initiated to develop such tools for most of the Indian language corpora. However, in recent years, a significant rise in number of Indian scientists getting associated with this kind of works has been noticed and such a positive phenomenon makes us happy with a promising aspiration and healthy expectation.

However, the shady side of this collective enterprise is that whatever tools are developed for the Indian language corpora, their rate of accuracy and efficiency is far below the level if these are compared with the corpus processing tools developed of English and other advanced languages of the world. Moreover – most of the time – tools that are claimed to be developed for one Indian language are not available for other Indian languages. Our tool is absolutely free from this limitation as it is freely available

for one and all. This indicates that we sincerely need to take collective initiatives develop good and useful corpus processing and accessing tools and devices for all Indian language corpora to analyze these corpora as well as retrieve information to be used in all spheres of language-related activity.

REFERENCES

- Anthony, L. 2004. AntConc: A learner and classroom friendly, multi-platform corpus analysis toolkit. In *Proceedings of WILL 2004: An Interactive Workshop on Language e-Learning* (pp. 7-13).
- Arnold, K. 2000. *The Java Programming Language*. New Delhi: Pearson Education.
- Biber, D. 1993. Using register diversified corpora for general language studies. *Computational Linguistics*, 2, 219-241.
- Bird, S. 2005. NLTK-Lite: Efficient scripting for natural language processing. In *Proceedings of 4th International Conference on Natural Language Processing* (pp. 1-8), IIT- Kanpur, India, Nov.
- . 2006. NLTK: The natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions* (pp. 69-72), Association for Computational Linguistics, Sydney, Australia, Jul.
- . 2008. Defining a core body of knowledge for the introductory computational linguistics curriculum. In *Proceedings of the 3rd Workshop on Issues in Teaching Computational Linguistics*, Association for Computational Linguistics.
- . & Loper, E. 2004. NLTK: The natural language toolkit. In *Companion Volume to the Proceedings of 42st Annual Meeting of the Association for Computational Linguistics* (pp. 214-217), Association for Computational Linguistics.
- , Klein, E., Loper, E. & Baldridge, J. 2008. Multidisciplinary instruction with the natural language toolkit. In *Proceedings of the Third Workshop on Issues in Teaching Computational Linguistics* (pp. 62-70), Association of Computational Linguistics.
- Bloch, J. 2005. *Java Puzzlers: Traps, Pitfalls and Corner Cases*. New Delhi: Pearson Education.
- BNC. Available online: <<http://corpus.byu.edu/bnc/>>.
- BNC. Available online: <www.natcorp.ox.ac.uk/docs/URG/index.html>.
- Brill, G. 2002. *CodeNotes for XML*. New York: Random House.
- Christ, O. 1994. A modular and flexible architecture for an integrated corpus query system. In *Proceedings of COMPLEX'94: 3rd*

- Conference on Computational Lexicography and Text Research* (pp 23-32), Budapest, Hungary, 7-10 Jul.
- Coding tutorials. Available online: <<http://www.w3schools.com/>>.
- Dash, N. S. 2007. Indian scenario in language corpus generation. In N. S. Dash, P. Dasgupta & P. Sarkar (Eds.), *Rainbow of Linguistics* (pp. 129-162). Vol. 1. Kolkata: T. Media Publication.
- Frath, P. & Gledhill, C. 2005. Free-range clusters or frozen chunks? Reference as a defining criterion for linguistic units. *Recherches Anglaises et Nord-américaines*, 38, 25-43.
- Hearst, M. 2005. Teaching applied natural language processing: Triumphs and tribulations. In *Proceedings of the Second ACL Workshop on Effective Tools and Methodologies for Teaching NLP and CL* (pp. 1-8), Association for Computational Linguistics, Ann Arbor, Michigan, Jun.
- Holmes, H. P. R., Ahmad, K. & Abidi, S. 1994. A description of texts in a corpus: Virtual and real corpora. In W. Martin, W. Meijs, M. Moerland, E. ten Pas, P. van Sterkenburg & P. Vossen (Eds.), *Proceedings of the 6th EURALEX International Congress on Lexicography* (pp. 390-402), Amsterdam, The Netherlands.
- Klein, E. 2006. Computational semantics in the natural language toolkit. In *Proceedings of the Australasian Language Technology Workshop (ALTW2006)* (pp. 26-33).
- Robinson, S., Aumann, G. & Bird, S. 2007. Managing fieldwork data with toolbox & the natural language toolkit. *Language Documentation and Conservation*, 1, 44-57.
- Shannon, C. 2003. Another breadth-first approach to CSI using Python. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education* (pp. 248-251).
- Sinclair, J. 2004. Intuition and annotation-the discussion continues. In K. Aijmer & B. Altenberg (Eds.), *Advances in Corpus Linguistics: Papers from the 23rd International Conference on English Language Research on Computerized Corpora (ICAME 23)* (pp. 39-59). Amsterdam/New York: Rodopi.
- Vaughan, L. & M. Thelwall. 2004. Search engine coverage bias: evidence and possible causes. *Information Processing and Management*, 40/4, 693-707.
- Wallis, S. & Nelson, G. 2001. Knowledge discovery in grammatically analysed corpora. *Data Mining and Knowledge Discovery*, 5, 307-340.

ACKNOWLEDGEMENT

I gladly acknowledge the technical support I received from Manomita Das and Kalpendu Das, two B. Tech students of the Bengal College of Engineering and Technology, Durgapur, West Bengal, for properly carrying out the work of developing the tools described above on the Bangla text corpus.

DR. NILADRI SEKHAR DASH
LINGUISTIC RESEARCH UNIT,
INDIAN STATISTICAL INSTITUTE, INDIA.
E-MAIL: <NS_DASH@YAHOO.COM>
PH: 91-9433190295