Checking the Correctness of Bangla Words using N-Gram

Nur Hossain Khan
B.Sc & M.Sc.
Dept. of CSE,
Islamic University,
Kushtia, Bangladesh.

Gonesh Chandra Saha
Asst. Professor
Dept. of CSIT,
Bangabandhu Sheikh Mujibur Rahman Agricultural University,
Gazipur, Bangladesh.

Bappa Sarker
Lecturer,
Dept. of CSE,
Islamic University,
Kushtia, Bangladesh.

Md. Habibur Rahman
Lecturer,
Dept. of CSE,
Islamic University,
Kushtia, Bangladesh.

ABSTRACT
N-gram model is used in many domains like spelling and syntactic verification, speech recognition, machine translation, character recognition and like others. This paper describes a system for checking the correctness of a bangle word using N-gram model. An experimental corpus containing one million word tokens was used to train the system. The corpus was a part of the BdNC01 corpus, created in the SIPL lab. of Islamic university. Collecting several sample text from different newspapers, the system was tested by 50,000 correct and another 50,000 incorrect words. The system has successfully detected the correctness of the test words at a rate of 96.17%. This paper also describes the limitations of the system with possible solutions.

General Terms
Artificial Intelligence, Natural Language Processing

Keywords
N-gram, Tokens, Corpus, Witten-Bell smoothing

1. INTRODUCTION
N-grams [1] are sequences of characters or words extracted from a text. N-grams can be divided into two categories: 1) character based and 2) word based. A character N-gram is a set of *n* consecutive characters extracted from a word. The main motivation behind this approach is that similar words will have a high proportion of N-grams in common. Typical values for *n* are 2 or 3; these correspond to the use of bigrams or trigrams, respectively. For example, the word *computer* results in the generation of the bigrams

C, CO, OM, MP, PU, UT, TE, ER, R

and the trigrams

C, *CO, COM, OMP, MPU, PUT, UTE, TER, ER*, R

where '*' denotes a padding space. There are *n*+1 such bigrams and *n*+2 such trigrams in a word containing *n* characters. Character based N-grams are generally used in measuring the similarity of character strings. Spellchecker, stemming, OCR error correction are some of the applications which use character based N-grams [2]. Word N-grams are sequences of *n* consecutive words extracted from text. Word level N-gram models are quite robust for modeling language statistically as well as for information retrieval without much dependency on language. N-gram based modeling finds extensive acceptance to the researchers working with structural processing of natural language. The idea of using n-grams in language processing was discussed first by Shannon [4]. An *n*-gram model is a type of probabilistic model for predicting the next item in such a sequence. More concisely, an *n*-gram model predicts x_i based on

$$X_{i-1}, X_{i-2}, X_{i-3}, \ldots\ldots\ldots, X_{i-n}.$$

In Probability terms, this is nothing but $P(X_i \mid X_{i-1}, X_{i-2}, \ldots\ldots\ldots, X_{i-n})$. An *n*-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram", size 3 is a "trigram"; and size 4 or more is simply called an "*n*-gram". For a sequence of words, for example "the dog smelled like a skunk", the trigrams would be: "# the dog", "the dog smelled", "dog smelled like", "smelled like a", "like a skunk" and "a skunk #". The idea of n-gram based word structure verification has come from these opportunities provided by n-grams. Word structure verification is the task of testing the syntactical correctness of a word. It is mostly used in word processors and compilers. For applications like compiler, it is easier to implement because the vocabulary is finite for programming languages but for a natural language it is challenging because of infinite vocabulary. In our work, an effort has been made to develop a system to verify Bangla word structure using statistical or more specifically n-gram based method. This is because, this approach does not need language resources like handcrafted grammatical rules, except for a corpus to train the system. Given the scarcity of language resources for Bangla, proposed approach may be the only reasonable one for the foreseeable future. One main advantage of the n-gram method is that it is language independent.[5]

2. TECHNIQUES ADOPTED IN THE PROPOSED SYSTEM
In statistical approach the probability of a word using n-gram analysis can simply be measured. For example, using bigram model probability of the word "বাংলা" is,

P ("বাংলা") = P (বা | <s>) * P (াং | বা) * P লা | াং) ∗ P (<s> | লা)

To estimate the correctness of a word, the probability of a word can be calculated using the formula above. If the value of the probability is greater than some threshold then the word is considered to be correct. Now if any of this character sequence (বা, াং, লা) is not in the corpus then the probability of the word will become zero because of multiplication. To solve this problem, Witten-Bell smoothing [3] was used to calculate the probability of a word in this research work. A sample corpus was used in this work that is a part of another corpus under construction in the speech and image processing lab of Islamic University, Bangladesh. Necessary programs were developed to assemble sequences of N tokens into N-

grams. Typically N-grams are formed of contiguous tokens that occur one after another in the input corpus.

If a bangla word "বা ং লা দে শ" is considered the possible bigrams (N-grams with N=2) are: <s> বা, বা ং, ং লা, লা দে, দে শ, শ <s>

Bigram probability, P(বাংলাদেশ) = P(বা | <s>) * P(ং | বা) * P(লা | ং) * P(দে | লা) * P(শ | দে)* P(<s> | শ)

and possible trigrams (Ngrams with N=3) are:

<s1><s2> বা, <s1> বা ং, বা ং লা, ং লা দে, লা দে শ, দে শ <s1>, শ <s1> <s2>

Trigram probability, P(বাংলাদেশ) = P(বা | <s1><s2>) * P(ং | <s1> বা) * P(লা | বা ং) * P(দে | ং লা) * P(শ | লা দে) * P(<s1>| দে শ)* P(<s2> | শ <s1>)

Similarly, the possible quad-grams (N-grams with N=4) are:

<s1><s2><s3> বা, <s1> <s2> বা ং, <s1> বা ং লা, বা ং লা দে, ং লা দে শ, লা দে শ <s1>,দে শ <s1> <s2>, শ <s1> <s2> <s3>

Quad-gram probability, P(বাংলাদেশ) = P(বা | <s1><s2><s3>) * P(ং | <s1> <s2> বা) * P(লা | <s1> বা ং) * P(দে | বা ং লা) * P(শ | ং লা দে)* P(<s1> | লা দে শ)* P(<s2> | দে শ <s1>)* P(<s3> | শ <s1> <s2>)

After training a model using above concept it was used to design a test system. For the purpose of testing whether a word is correct or not, the number of N-grams (2, 3, or 4) in the word was counted first. Using all the N-grams of the word a score for the word is generated. If the score is greater than a predefined threshold, the word is syntactically correct. On the other hand, if the score is not greater than the threshold, the word is syntactically incorrect.

3. TRAINING THE N-GRAM MODEL

The first step to compute N-grams is counting unigrams. The unigram count and necessary software tools was ready in the laboratory and the work was started from bigram count. After updating the existing software tools bigrams, trigrams and quad-grams were identified, counted and stored in separated disk files. In all cases input to the software was the sample corpus contained in file corpus.txt. The outputs are shown in figure-1(a) & 1(b)

আবার জমজমাট রাজনীতি।
দীর্ঘদিনের জড়তা কাটিয়ে
ফের সরগরম রাজপথ।
আবার জমজমাট রাজনীতি।
.....
.....

Unigram	Frequency
আ	১০০০০০০
বা	২৫৬০০০
র	১৫০২০৫৪
জ	৩৪৯১৫৪৩
ম	২৮৬১৯৭৬
জ	৪৩৮১৩৪২
মা	১১৪২৪৫৩
ট	১৭৭৮৪৩২

Figure 1(a): Samples of first step computation

Bigram	Frequency
আ বা	৩০৬০২৭৪
বা র	৬৪৫৭৮৮৪
জ ম	২৫৩৪৮৮৪
ম জ	২৩৫৬৬৬৭
জ মা	৫৬২৩৪৭৩
মা ট	৩৬৭৩৪৬৫

Trigram	Frequency
আ বা র	২৩৪৭৭৩
জ ম জ	১২৩৪৮৭
ম জ মা	২২৩১৪৫
জ মা ট	২১৩৪৪৫

Quadrigram	Frequency
জ ম জ মা	১০০৩
ম জ মা ট	০৯৫০

Figure 1(b): Samples of first step computation

In the second step of computation, outputs of the first step were used as inputs. A set of program modules were developed to compute bigram, trigram and quad-gram probabilities using N and N-1 gram count. For example, bigram probabilities were calculated by using unigram and bigram counts. The intermediate results of the system as the outputs of the second step are shown in figure-2.

Bigram	Probability
আ বা	০.৯৮
বা র	০.৭৮
জ ম	০.৭২
ম জ	০.৬৭
জ মা	০.৮৮
মা ট	০.৭৯

Trigram	Probability
আ বা র	০.৯৩
জ ম জ	০.৮২
ম জ মা	০.৭৯
জ মা ট	০.৯৮

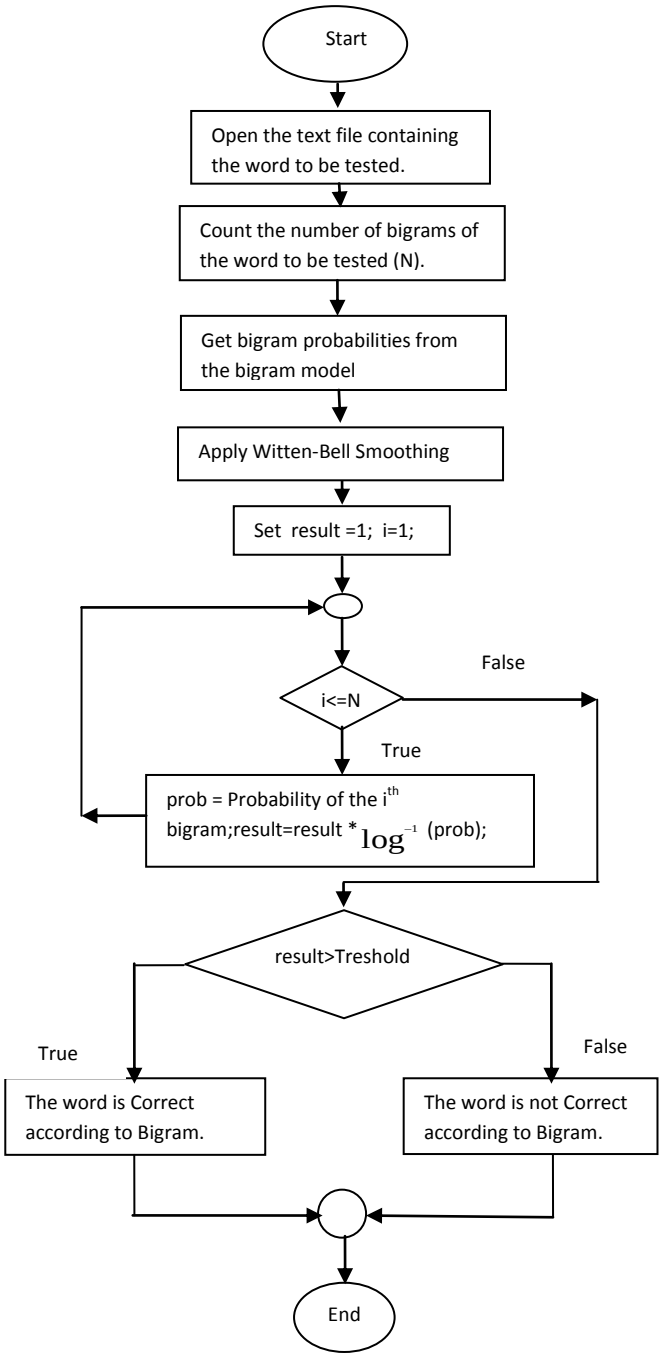
Quadrigram	Probability
জ ম জ মা	০.৭১
ম জ মা ট	০.৬৬

Figure-2: Sample results of second step computation

4. THE TEST SYSTEM

For the purpose of testing whether a word is correct or not, at first, all the number of bigrams of the word was counted. Getting probabilities from the respective models, Witten-Bell smoothing was applied to compute a set of probabilities

contained all nonzero values. Multiplying all the bigrams of the word, a score for the word was generated. If the score is greater than a predefined threshold, the word is syntactically correct. The functional block diagram of the system is shown in figure 3. For the trigram or quad-gram models, the same algorithm was followed by replacing only the bigrams with trigrams or quad-grams respectively.



5. EXPERIMENTAL RESULTS AND DISCUSSION

In our experiment, 50,000 words were collected from the web edition of several daily newspapers for the test set. The test set was disjoint from the training corpus. All of these 50,000 words were structurally correct. Taking these correct words as input, the result generated by the test system is shown in table–1. For another experiment, All of these 50,000 words were modified in different ways to make them incorrect and presented again as input to the test system. The result generated by second experiment is also shown in table-1.

6. DISCUSSION

The developed system can detect the correct words as correct at the rate of 95.19% and incorrect words as incorrect at the rate of 97.14%. Therefore the average performance of the system is 96.17%.

Table-1: The Test Result with Correct And Incorrect Words

Results with correct sentences			
Models	*No. of words*	*No of success*	*Performance*
Bigram	50000	47600	95.20%
Trigram	50000	47560	95.12%
Quadrigram	50000	47630	95.26%
Results with incorrect sentences			
Bigram	50000	48565	97.13%
Trigram	50000	48570	97.14%
Quadrigram	50000	48581	97.16%
		Average	96.17%

7. CONCLUSION

We have developed a statistical word verifier for Bangla language, which has a reasonably good performance as a rudiment word verifier. The error rate of our system is 3.83%. The major cause of this error is the volume of training corpus. By increasing the volume of training corpus the overall performance of the system can be increased. This research work is a preliminary task of Bangla Spell Checker. By adding some functionality this task can be converted to Bangla Spell Checker which is the future plan of this research.

8. REFERENCES

[1] P Majumder, M Mitra, B.B. Chaudhuri, "N-gram: a language independent approach to IR and NLP", ICUKL November 2002, Goa, India.

[2] Wikipedia, "n-gram", *http://en.wikipedia.org/wiki/N-gram*, Access date: 17th Dec. 2013.

[3] Daniel Jurafsky, James H. Martin,"Speech and Language Processing An Introduction to Natural Language Processing: Computational Linguistics and Speech Recognition", Prentice Hall, Englewood Cliffs, New Jersey 07632 , September 28, 1999

[4] C. E. Shannon, "Prediction and entropy of printed English," Bell Sys. Tec. J. (30):50–64, 1951

[5] Farag Ahmed, Ernesto William De Luca, and Andreas Nürnberger, "Revised N-Gram based Automatic Spelling Correction Tool to Improve Retrieval Effectiveness", August 22, 2009

[6] Hasan Muaidi, Rasha Al-Tarawneh, "Towards Arabic Spell-Checker Based on N-Grams Scores", International Journal of Computer Applications (0975 -8887), Volume 53 - No. 3, September 2012.