History (Forward N-Gram) or Future (Backward N-Gram)? Which Model to Consider for N-Gram Analysis in Bangla?

Naira Khan, Md. Tarek Habib, Md. Jahangir Alam, Rajib Rahman, Naushad UzZaman and Mumit Khan

Center for Research on Bangla Language Processing, BRAC University, Bangladesh

naira@bracu.ac.bd, md.tarekhabib@yahoo.com, jahangir_bu@yahoo.com, rajib77bd@yahoo.com, naushad@bracu.ac.bd, mumit@bracu.ac.bd

Abstract

This paper presents a directional advantage of n-gram modeling in terms of backward or forward n-gram modeling in Bangla. The most commonly used n-gram analysis is predominantly a forward n-gram. However in Bangla it appears that a backward n-gram is repeatedly more successful and yields more grammatical results than a forward n-gram. This paper hypothesizes that the rationale behind this success is the syntactic ordering of constituents in Bangla. Bangla is a head-final specifier-initial language as opposed to English, which is head-initial specifier-initial. Hence in Bangla, the head comes after its argument in a phrase. If an n-gram analysis begins with a head and moves backwards it will stretch to its own argument but if you move for-wards then you'll probably grab the argument of an-other head. As probability of occurrence of heads is higher, probability of depending on a head is also higher and hence a backward n-gram will probably have a greater chance of yielding grammatical results. We carried out several experiments to compare different directional results in different applications with an ad-vantage in the backward direction. This will prove a useful linguistic insight in terms of n-gram based analysis depending upon variations of constituent analysis.

1. Introduction

An n-gram is a sub-sequence of n items in any given sequence. In computational linguistics n-gram models are used most commonly in predicting words for the purpose of various applications. The use of n-grams for such purposes is known as language modeling (LM), the field of modeling on how text is generated and recognized [1]. In such analysis, a "likelihood" value is assigned to a given string of words. For example, the string ''he went home'' is more likely than ''abacus kindly flew'', so the previous string will be assigned a higher likelihood value than the latter. A typical application of this kind of analysis is speech recognition, where a language model can help the system rank a set of candidate sentences by measuring the likelihood of their utterances.

2. Forward N-Gram vs. Backward N-Gram

Formally, we consider a string of words $W = w_1 . . .w_n$. We are interested in creating an expression $P (W) = P (w_n \mid w_1 .....w_{n-1})$ - a probability distribution over the vocabulary set (of size |V|), given the history of words. And for backward n-gram $P (W) = P (w_k \mid w_{k+1} … w_{k+n-1})$. Given these language models, the "likelihood" of a string of words can be calculated as P (W).

In a forward n-gram, the probability of each word is estimated depending on the preceding word. In other words, the n-gram analysis moves in a forward direction where the prediction depends on the history. On the other hand, in a backward n-gram the probability of each word is estimated depending on the following words, where the prediction depends on the future.

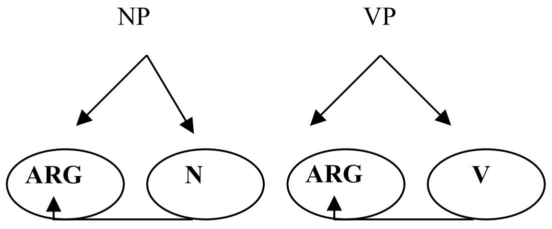
Conventionally, the forward n-gram method is used most predominantly for language modeling. However, we have experimentally found that a backward n-gram yields better results in various applications for Bangla. We present these findings and hypothesize the reason behind this directional advantage in the following sections.

3. Hypothesis

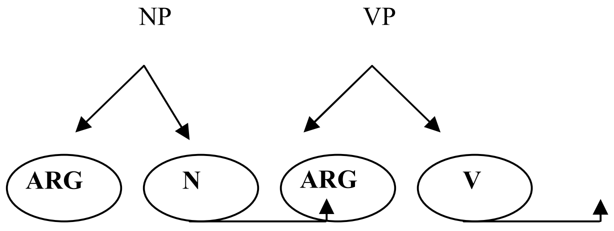
We hypothesize that a backward n-gram works better than forward n-gram because Bangla is a head-final language. In other words, in a Bangla phrase (e.g., in a noun, verb, or postpositional phrase), the head comes after its argument or is in the final position. In case of a noun-phrase the head is the noun and its argument is the specifier (minimally a determiner), and for verb phrases, the head is the verb and the argument is the complement of the verb. Bangla is a head-final and specifier-initial language as opposed to English, which is head-initial and specifier-initial. Since an argument can't exist without the head, it follows that any body of Bangla text will contain the sequences [+argument +head] or [-argument +head] but never the sequence [+argument -head]. So, in general, heads will occur more frequently than arguments. If an n-gram analysis depends on the head then moving backwards will combine the head to its argument. However, moving forward will combine it with the argument of another head. If, however, the n-gram is based on the argument in the first place, then moving forward will provide grammatical coherence rather than moving backwards. As probability of occurrence of heads is higher, probability of n-gram based on a head is also higher and hence a backward n-gram has a greater chance of yielding grammatical results.

The Phrase Structure (PS) rules for Bangla are
S -> NP VP
NP -> ARG N
VP -> ARG V

Backward n-gram



Forward n-gram



4. Analysis

Language modeling using backward n-gram contains information that is complementary to the information in the language modeling using forward n-gram.

Our hypothesis is tested by experimenting in three types of applications. They are as follows:
- Grammar checking
- Parts of Speech (POS) tagging
- Sentence generation

The experiments along with their analysis are given below.

4.1. Grammar checking

A Grammar checker determines the syntactical correctness of a sentence. Three methods are widely used for grammar checking in a language: syntax based parsing, statistical approach and rule based approach. In syntax based grammar checking [2], each sentence is completely parsed to check the grammatical correctness of it. The text is considered incorrect if the parsing does not succeed. In statistics-based approach [3], a POS-annotated corpus is used to build a list of POS tag sequence. Some sequences will be very common (for example, *determiner, adjective, noun* as in '*the old man*'), others will probably not occur at all (for example, *determiner, determiner, adjective*). Uncommon sequences in the training corpus can be considered incorrect in this approach. In a rule based approach [4], a set of hand crafted rules is matched against a text which has at least been POS tagged. This approach is very similar to a statistics-based approach, but the rules are developed manually.

For Bangla we developed a statistical grammar checker based on n-gram analysis (both forward and backward n-gram) of words.

For example, in forward bigram (considers history), the probability of the sentence "He is playing." is:
P ("He is playing") = P (He | <start>) * P (is | He) * P (playing | is) * P (. | playing)
On the contrary, in backward bigram (considers future), the probability of the sentence "He is playing." is:
P ("He is playing") = P (He | is) * P (is | playing) * P (playing | .) * P(. | <end>)

To estimate the grammatical correctness of an n-gram based grammar checker, we calculate the probability of a sentence using the formula above. If the value of the probability is above some threshold then we consider the sentence to be grammatically correct.

Now if any of these three words (He, is, playing) are not in the corpus then the probability of the sentence will become zero because of multiplication.

In our calculations, we calculated the probability using this general n-gram technique; we also used two other smoothing techniques [1]: add-one smoothing and Witten-Bell smoothing, to calculate the probability of a sentence.

We trained our n-gram model (both forward and backward for bigram and trigram models) with a 39357 token-sized corpus of The Daily Prothom-Alo [5]. We have experimented with 50 sentences extracted from the same newspaper, but the test set is disjoint from the training corpus. Among these 50 sentences, 30 sentences were grammatically correct and we modified 20 other sentences to make those sentences grammatically incorrect. We have calculated the probability of all 50 sentences using add-one smoothing, Witten-Bell smoothing, and without any smoothing technique for bigram model and again calculated the probability using add-one smoothing and without any smoothing technique for trigram model.

For bigram model, our result suggested that without smoothing, backward n-gram performed better than forward n-gram. Backward n-gram detected 27 grammatically correct sentences out of 30 sentences. Using add-one smoothing, the backward model again performed better. But it detected all 50 sentences as correct sentence, where 20 grammatically incorrect sentences were present. Using Witten-Bell smoothing, forward n-gram detected 10 sentences to be correct, where 7 were correct; and backward n-gram detected 40 sentences to be correct, where 23 sentences were correct. So, again backward n-gram performed better than forward n-gram.

Table 1: Comparison between forward and backward bigram

BIGRAM RESULT		
Without smoothing	**Overall**	**Correct**
Forward	0	0
Backward	27	27
Add-one smoothing	**Overall**	**Correct**
Forward	0	0
Backward	50	30
Witten-Bell smoothing	**Overall**	**Correct**
Forward	10	7
Backward	40	23

For trigram model, our result suggested that without smoothing backward n-gram performed better than forward n-gram. Backward n-gram detected 14 grammatical correct sentences out of 30 sentences. Using add-one smoothing, backward model again performed better. But it detected all 50 sentences as correct sentence, where 20 grammatically incorrect sentences were present.

Table 2: Comparison between forward trigram and backward trigram

TRIGRAM RESULT		
Without smoothing	**Overall**	**Correct**
Forward	0	0
Backward	14	14
Add-one smoothing	**Overall**	**Correct**
Forward	0	0
Backward	50	30

Our experiment result suggested that for both bigram and trigram, backward n-gram suggests better result than forward n-gram, which consolidates our hypothesis.

4.2. POS Tagging

Part-Of-Speech (POS) tagging is a technique for assigning each word of a text with an appropriate parts of speech tag. The significance of parts-of-speech (also known as POS, word classes, morphological classes, or lexical tags) for language processing is the large amount of information they give about a word and its neighbor. POS tagging can be used in TTS (Text to Speech), information retrieval, shallow parsing, information extraction, linguistic research for corpora [6] and also can be used as an intermediate step for higher level NLP tasks, such as, parsing, semantics, translation, and many more [7], which make POS tagging a necessary application for advanced NLP applications in Bangla or any other languages.

We implemented a simple stochastic n-gram (forward and backward) based tagger for POS tagging. The intuition behind all stochastic taggers is a simple generalization of the "pick the most likely tag for this word" approach.

For a forward n-gram tagger, we calculate the probability of tag-sequence by P (tag | previous *n* tags) and calculate the probability of word likelihood by P (word | tag). Finally we multiply these two probabilities to check, for which tag it maximizes the probability.

Formula for forward n-gram POS tagger:

P (word| tag) * P (tag | previous *n* tags)

Backward n-gram POS tagger works same as forward n-gram POS tagger, except the case, it considers the next n tags rather than previous n tags.

Formula for backward n-gram POS tagger:
P (word | tag) * P (tag | next *n* tags)

In the experiment of POS tagging, a tagged corpus of about 3000 words from The Daily Prothom-Alo (Bangla) [5] and Brown corpus (English) [8] are used. We also experimented on bigram and trigram POS tagging model for both Bangla and English to see how both of these languages perform.

Our experiment resulted that for English, traditional forward n-gram POS tagger performed better than backward n-gram POS tagger.

Table 3: Performance for different n-gram in English

Forward n-gram		Backward n-gram	
Bi-gram	Tri-gram	Bi-gram	Tri-gram
72.2%	72.0%	71.7%	71.8%

Unlike English, backward bigram POS tagger performed better for Bangla, and trigram performed similarly for forward and backward taggers.

Table 4: Performance for different n-gram in Bangla

Forward n-gram		Backward n-gram	
Bi-gram	Tri-gram	Bi-gram	Tri-gram
67.6%	68.7%	67.9%	68.7%

From the experiment of POS tagging we see that the performances of forward and backward tagging in both Bangla and English differ slightly with a small advantage of backward n-gram for Bangla as opposed to English, where it appears that forward n-gram has better performance. The size of our corpus was 3000 words, which was too small to differentiate the two approaches. However, we can predict that for Bangla, backward tagging may perform better than forward tagging, even if the corpus size is increased.

4.3. Sentence generation

Sentence generation is a form of language generation. Its task is to generate sentence having maximum likelihood. In a sentence generation application using n-gram, seeing n-1 words we calculate which word is most probable to occur at nth position. This is basically what forward n-gram is, using the history of n-1 words to predict the nth word. On the other hand, backward n-gram uses the future n-1 words to predict what will be the current word.

Sentence generation using forward n-gram: $W = w_1, w_2, \ldots, w_{n-1}, \mathbf{w_n}$
Predict w_n, based on the probability of previous n-1 words, $w_1, w_2, \ldots, w_{n-1}$.

Sentence generation using backward n-gram: $\mathbf{w_1}, w_2, \ldots, w_{n-1}, w_n$
Predict w_1, based on the probability of next n-1 words, $w_2, \ldots, w_{n-1}, w_n$.

To generate sentence in forward n-gram based sentence generator, we need to input a starting word of the sentence and the model outputs the whole sentence based on n-gram probabilities. In case of backward n-gram based sentence generator, we need to input an ending word of the sentence and the model outputs the whole sentence based on n-gram probabilities.

We have generated sentences using forward and backward n-gram (n = 2, 3 and 4; i.e. bigram, trigram and quadrigram) model. In both models, if we increase the value of n the accuracy increases. For Bangla we have seen that quadrigram is more accurate than bigram and trigram and generate more likely sentences. From our experiment we have seen that backward n-gram based sentence generator outputs more grammatical sentences than forward n-gram based sentence generator. In the following sub-sections we have shown few examples of sentence generator.

Sentence generation output for forward n-gram
Starting word: ইরাকে
Forward Bigram: ইরাকে সদ্যসমাপ্ত টেস্টে শেবাগের দখলে থাকা পুলিশের সোর্স শাহজাহানের নাম কী ?
Forward Trigram: ইরাকে নিযুক্ত মার্কিন রাষ্ট্রদূত জালমে খলিলজাদ বলেন , এ ধরনের আহ্বানের মাধ্যমে আবদুর রহমানসহ তার দলের সব এহসার ও গায়রে এহসাররা সহজে স্বেচ্ছায় ধরা দিতে পারে .
Forward Quadrigram: ইরাকে নিযুক্ত মার্কিন রাষ্ট্রদূত জালমে খলিলজাদ বলেন , নির্বাচনে ভোটদান প্রক্রিয়া ও ফলাফল নিয়ে ইরাকি জনগণের আত্মবিশ্বাস রয়েছে .
Starting word: এক
Forward Bigram: এক পর্যায়ে .
Forward Trigram: এক পর্যায়ে মুকুল নিস্তেজ হয়ে পড়লে বাংলা ভাই তার ছেলেকে বাড়ি থেকে তুলে নিয়ে এই টানাপড়েনের কারণে শতাধিক শিক্ষার্থীর লেখাপড়া ব্যাহত হচ্ছে .
Forward Quadrigram: এক পর্যায়ে মুকুল নিস্তেজ হয়ে পড়লে বাংলা ভাই নিজে তাহেরপুর পুলিশ ফাঁড়িতে রেখে আসে তার মৃতপ্রায় দেহ.
Sentence generation output for backward n-gram

Bengali

End word is: হয়
Backward Bigram Sentence:হবে বলে তিনি বলেন , গত ২৬ ডিসেম্বর প্রায় ১ লাখ টাকা মূল্যের গমবীজ আটক করে থানায় মামলা দায়ের করা হয় .
Backward Trigram Sentence:নীলিমা এর বেশি আর কিছু জানেন না আসলে সংস্থাটির কর্মকর্তারা প্রায় ১০ লাখ টাকা উদ্ধার করা হয় .
Backward Quadrigram Sentence:একই মাসে বারুইহাটি গ্রামের শহিদুল ইসলাম নামে এক যুবককে হরণের পর হত্যা করা হয় .
End word is: হয়েছে
Backward Bigram Sentence:হবে বলে তিনি বলেন , গত ২৬ ডিসেম্বর প্রায় ১ লাখ টাকা মূল্যের গমবীজ আটক করে থানায় মামলা দায়ের করা হয়েছে .
Backward Trigram Sentence:পুলিশ প্রথমে তাহেরপুর স্বাস্থ্যকেন্দ্রে এবং পরে রাজশাহী মেডিকেল কলেজ হাসপাতালে ভর্তি করা হয়েছে .
Backward Quadrigram Sentence:এদের পাকশী রেল হাসপাতালে ভর্তি করা হয়েছে .

5. Future Work

This paper may prove useful as a linguistic basis for n-gram advantage in head-final or head-initial languages for performance optimization. However, it must be mentioned that a strong claim for the hypothesis proposed in this paper cannot be made due to lack of data as the experiments were small scale and only three applications were tested. In order to make concrete our hypothesis an interesting future endeavor would be to run a large-scale analysis as well as a comparison of performance results in head-initial and head-final languages.

6. Conclusion

N-grams are used very commonly in many different NLP applications. Most commonly a forward n-gram is used rather than a backward n-gram. However, it appears that the backward n-gram yields better results in Bangla than a forward n-gram, which in turn performs better in English. This paper attempts to show that the directional advantage of n-grams may not be arbitrary in that there may be a sound linguistic basis for one to perform better than the other. Although the experiments presented here were small scale, however, it appears that a backward n-gram repeatedly has an advantage over a forward n-gram for Bangla and vice versa in English. Our linguistic hypothesis states that this difference in performance is based on the differing constituent ordering of the two languages as Bangla is head-final and English is head-initial. This paper may prove to be a starting point in an endeavor to conduct a large scale analysis in various applications and parallel comparison run on languages with different constituent ordering in order to take this hypothesis further and thus prove useful in optimizing the performance of n-gram based applications.

7. Acknowledgment

This work has been supported in part by the PAN Localization Project (www.panl10n.net) grant from the International Development Research Center, Ottawa, Canada, administrated through Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences, Pakistan.

8. References

[1] D. Jurafsky and J.H. Martin, *Speech and Language Processing*, Prentice Hall, 2000.

[2] K. Jensen, G.E. Heidorn, S.D. Richardson (Eds.), *Natural Language Processing, the PLNLP approach,* 1993.

[3] E. Atwell and S. Elliott, *Dealing with ill-formed English text, The Computational Analysis of English*, Longman, 1987.

[4] D. Naber, *A Rule-Based Style and Grammar Checker*, Diploma Thesis, Computer Science - Applied, University of Bielefeld, 2003.

[5] Bangladeshi Newspaper, Prothom-Alo. Online version available online at: http://www.prothom-alo.net/.

[6] D. Jurafsky and J.H. Martin, *Speech and Language Processing*, Prentice Hall, 2000.

[7] Y. Halevi, "Part of Speech Tagging", *Seminar in Natural Language Processing and Computational Linguistics (Prof. Nachum Dershowitz)*, School of Computer Science, Tel Aviv University, Israel, April 2006.

[8] Brown Tagset, available online at: http://www.scs.leeds.ac.uk/amalgam/tagsets/brown.html.