

## **N-gram: a language independent approach to IR and NLP**

P Majumder, M Mitra, B.B. Chaudhuri  
Computer vision and pattern recognition Unit  
Indian Statistical Institute, Kolkata  
mandar@isical.ac.in

### **Abstract**

*With the increasingly widespread use of computers & the Internet in India, large amounts of information in Indian languages are becoming available on the web. Automatic information processing and retrieval is therefore becoming an urgent need in the Indian context. Moreover, since India is a multilingual country, any effective approach to IR in the Indian context needs to be capable of handling a multilingual collection of documents. In this paper, we discuss the N-gram approach to developing some basic tools in the area of IR and NLP. This approach is statistical and language independent in nature, and therefore eminently suited to the multilingual Indian context. We first present a brief survey of some language-processing applications in which N-grams have been successfully used. We also present the results of some preliminary experiments on using N-grams for identifying the language of an Indian language document, based on a method proposed by Cavnar et al [1].*

### **1. Introduction**

N-grams 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.

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.

#### **1.1 N-gram based language modeling**

Informally speaking, a language is modeled by making use of linguistic and common sense knowledge about the language. Formally, a language model is a probability distribution over word sequences or word N-grams. Specifically, a language model (LM) estimates the probability of next words given preceding words. A word N-gram language model uses the history of  $n-1$  immediately preceding words to compute the occurrence probability  $P$  of the current word. The value of  $N$  is usually limited to 2 (bigram model) or 3 (trigram model). If the vocabulary size is  $M$  words, then to provide complete coverage of all possible  $N$  word sequences the language model needs to consist of  $M^N$ -grams (*i.e.*, sequences of  $N$  words). This is prohibitively expensive (*e.g.*, a bigram language model for a 40,000 words vocabulary will require  $1.6 \times 10^9$  bigram pairs), and many such sequences have negligible probabilities. Obviously, it is not possible for an N-gram language model to estimate probabilities for all possible word pairs. Typically an N-gram LM

lists only the most frequently occurring word pairs, and uses a backoff mechanism to compute the probability when the desired word pair is not found.

For instance, in a bigram LM, given  $w_i$ , the probability that the next word is  $w_j$  is given by:

$$\hat{p}(w_j|w_i) = \begin{cases} p(w_j|w_i) & (w_i, w_j) \text{ exists} \\ b(w_i)p(w_j) & \text{otherwise} \end{cases}$$

where  $b(w_i)$  is the back-off weight for the word  $w_i$ ,

$p(w_i)$  is the unigram probability of the  $w_i$

The backoff weight  $b(w_i)$  is calculated to ensure that the total probability:

$$\sum_j \hat{p}(w_i, w_j) = 1$$

Similarly, for a trigram of words  $w_h w_i w_j$ ,

$$\hat{p}(w_j|w_h w_i) = \begin{cases} p(w_j|w_h w_i) & (w_h, w_i, w_j) \text{ exists} \\ b(w_h w_i) \hat{p}(w_j|w_i) & \text{otherwise} \end{cases}$$

The rest of the paper is organized as follows. Section 2 presents a brief survey of applications of the N-gram approach to language-related problems. Section 3 describes a small experiment we did for language identification using N-grams, based on a method proposed by Cavnar et al[1]. We have tested the character level N-gram algorithms for language identification from a multilingual collection of Indian language documents. Finally, Section 4 outlines some future directions of working with N-grams in the Indian context.

## 2. N-gram applications

Speech recognition, handwriting recognition, information retrieval, optical character recognition, spelling correction and statistical stemmers are some major areas where “N-gram” based statistical language modeling can play an important role.

Character “N-gram” matching for computing a string similarity measure is widely used technique in information retrieval, stemming, spelling and error correction [5-11], text compression [12], language identification [13-14], and text search and retrieval [15-16]. The N-gram based similarity between two strings is measured by Dice’s coefficient. Consider the word *computer* whose bi-grams are :

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

To measure the similarity between the words *computer* and *computation*, we can use Dice’s coefficient in the following way. First, find all the bi-grams from the word *computation*

\*C, CO, OM, MP, PU, UT, TA, AT, TI, IO, ON, N\*

The number of unique bi-grams in the word *computer* is 9 and in the word *computation* is 12. There are 6 common bi-grams in both the words. Similarity measured by Dice's coefficient is calculated as  $2C/(A+B)$ , where A and B are the number of unique bigrams in the pair of words; C is the number of common bigrams between the pair. For statistical stemming, terms are clustered using the "Single link Clustering Method" along with the above similarity measure. For spelling correction tri-gram matching gives significant results [2]. Some IR systems [20] use character N-grams rather than words as index terms for retrieval is done, and the system works unmodified for documents in English, French, Spanish, and Chinese. The resilience provided by character N-grams against minor errors in the text was an advantage for this system.

Categorization of text into some preexisting categories is another fundamental need for document processing. Cavnar and Trenkle proposed a method for N-gram based language identification and text categorization in English [1]. Furnkranz [19] showed results with a rule learning algorithm that indicate that, after the removal of stop words, word sequences of length 2 or 3 are most useful. Using longer sequences reduces classification performance. Damashek [17] proposes a simple but novel vector-space technique that makes sorting, clustering and retrieval feasible in a large multilingual collection of documents. The technique only collects the frequency of each N-gram to build a vector for each document and the processes of sorting, clustering and retrieval can be implemented by measuring the similarity of the document vectors. It is language-independent. A little random error only influences a small quantity of N-grams and will not change the total result. This method thus provides a high degree of robustness.

Tan et al.[3] propose a method of text retrieval from document images using a similarity measure based on an N-gram algorithm. They directly extract image features instead of using optical character recognition. Character image objects are extracted from document images based on connected components first and then an unsupervised classifier is used to classify these objects. All objects are encoded according to one unified class set and each document image is represented by one stream of object codes. Next, they retrieve N-gram slices from these streams and build document vectors and obtain the pair-wise similarity of document images by means of the scalar product of the document vectors.

In case of speech and handwriting recognition, word N-grams help the computer to resolve ambiguities among different linguistic constituents in given contexts. Zhao [4] investigates the efficiency of implementing the N-gram decoding processes in speech recognition.

A tri-gram Language Model has also been successfully used for speech recognition by L. Bahl et al[18]. In general, for a given word sequence  $W=\{w_1, \dots, w_n\}$  of  $n$  words, the LM probability is:

$$p(W) = p(w_1, \dots, w_n) = \prod_{i=1}^n p(w_i | w_0, \dots, w_{i-1})$$

where  $w_0$  is chosen appropriately to handle the initial condition. The probability of the next word  $w_i$  depends on the history  $h_i$  of words that have been spoken so far. With this factorization the complexity of the model grows exponentially with the length of the history. To have a more practical and parsimonious model, only some aspects of the history are used to affect the probability of the next word. Specifically, Bahl et al. use the trigram model. The probability of a word sequence under this model becomes:

$$p(W) \approx \prod_{i=1}^n p(w_i | w_{i-2}, w_{i-1}).$$

A large text corpus (training corpus) is used to estimate trigram probabilities. These probabilities then correspond to trigram frequencies as follows:  $p(w_3 | w_1, w_2) = c_{123} / c_{12}$  Where  $c_{123}$  is the number of times the sequence of words  $\{w_1, w_2, w_3\}$  is observed and  $c_{12}$  is the number of times the sequence  $\{w_1, w_2\}$  is observed.

For a vocabulary size  $K$  there are  $K^3$  possible trigrams. For example a vocabulary of 20,000 words means 8 trillion trigrams. But many of these trigrams will not appear in the training corpus. So the probabilities of unseen trigrams should be smoothed. This can be done by linear interpolation of trigram, bigram, and unigram frequencies and a uniform distribution on the vocabulary.

One of the major problems of n-gram modeling is its size. For a Vocabulary of 20,000 words number of bigrams = 400 million, number of trigrams = 8 trillion, number of four-grams =  $1.6 \times 10^{17}$ . so the number of indexing unit will increase enormously.

### 3. Our experiment

In our experiment, we followed the algorithm proposed by Cavnar et.al[1] for identifying Indian languages from a multilingual collection of documents. We first create character N-gram profiles for 10 Indian languages. The N-gram frequency profile is generated by counting all the N-grams in a set of documents in particular language, and sorting them in descending order. The maximum occurring N-grams are monograms and they occur at the top of the list; then comes the bi-grams and tri-grams. We calculated frequencies up to penta-grams as proposed by Cavnar et al.

When a new document whose language is to be identified comes, we first create an N-gram profile of the document and then calculate the distance between the new document profile and the language profiles. The distance is calculated according to “*out-of-place measure*” between the two profiles. The shortest distance is chosen and it is predicted that the particular document belongs to that language. A threshold value has been introduced so that if any distance goes above the threshold, then the system claims that the language of the document cannot be determined.

For categorization, character N-gram profiles of several predetermined categories were created. The N-gram profile of the new document is prepared and the distance is measured using the same algorithm used for language identification and within the limit of a predetermined threshold. We prepare language profile by using the TDIL corpus. We first choose 100 documents from each of the language for making the N-gram language profile. Again the shortest distance is chosen and the prediction goes in favor of the shortest distance category.

Indian languages can be grouped into five categories based on their origin:

1. Indo-European (Hindi, Bangla, Marathi, etc.)
2. Dravidian (Tamil, Telegu, etc.)
3. Tibeto-Burmese (e.g. Khasi)
4. Astro-Asiatic (Santhali, Mundari, etc.)
5. Sino-Tibetan (e.g. Bhutanese)

Languages within a group share a number of common elements. For instance, there is a significant overlap in the vocabulary of Bangla and these languages will be closer than the profiles for a pair of languages from two different groups. In our current experiment we find difficulties to distinguish between urdu and hindi language documents. as hindi and urdu both share a common vocabulary a considerable amount of indexing unit is common for both the language.

Table1 shows some sample results where the language of the test document is identified according to the least distance measured between the language profile and the document profile.

Presently we are testing the system for document categorization. Our recent finding states that Profiles generated by character level N-gram and word level N-gram gives much better results in both the cases. Work about Exploring the potentialities of N-gram in case of Indian Language is in progress.

We have calculated all possible distances between each language profiles. Below the matrix showed gives the distances. The most frequent top 5000 n-grams are considered from each language profile for

calculating all possible distance between the languages. Table 2 shows the distances between all the profiles in order of  $10^6$ . The diagonal is zero. The matrix is symmetrical. The nearness of a language with the other can be discovered from the matrix

### Language Identification

Name of File	Bangla	Hindi	Tamil	Urdu	Conclusion
/cdrom/tdil/bengali/1000	1147361422	7956588958	1804097714	16255858411	Bangla
/cdrom/tdil/hindi/201	1266565118	1226658566	1422694309	13110566420	Hindi
/cdrom/tdil/urdu/aerol	9573773827	9784027400	1192113150	0474097877	Urdu
/cdrom/tdil/Tamil/101	6651174543	10635333243	2339901208	20870156044	Tamil

Table 1.

PROFILE	Bangla	Hindi	Kannada	Kashmiri	Malayalam	Telugu	Urdu
Bangla	0	16.54	19.42	23.66	20.27	19.08	24.01
Hindi	16.54	0	18.40	23.65	19.74	18.58	24.12
Kannada	19.42	18.40	0	23.80	18.11	16.65	24.09
Kashmiri	23.66	23.65	23.80	0	24.02	23.88	19.54
Malayalam	20.27	19.74	18.11	24.02	0	18.07	24.29
Telugu	19.08	18.58	16.65	23.88	18.07	0	24.15
Urdu	24.01	24.12	24.09	19.54	24.29	24.15	0

Table 2.

**4. Future direction:** The statistical and language independent nature of N-gram model seems suitable for dealing with a multilingual collection of texts. Improving retrieval efficiency from Indian language documents by N-gram will be our future effort. We will also investigate the use of N-gram language modeling for Information retrieval, text categorization and machine translation.

### References:

- [1] William B. Cavnar, John M. Trenkle "N-Gram-Based Text Categorization" Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval

- [2] R. Anglell, G. Freund, and P. Willett, "Automatic spelling correction using a trigram similarity measure", *Information Processing & Management*, 19, (4), 305--316, (1983).
- [3] Chew Lim Tan, Sam Yuan Sung, Zhaohui Yu, and Yi Xu "Text Retrieval from Document Images based on N-gram Algorithm" PRICAI Workshop on Text and Web Mining.  
<http://citeseer.nj.nec.com/400555.html>
- [4] Jie Zhao, "NETWORK AND N-GRAM DECODING IN SPEECH RECOGNITION" Masters Thesis  
Department of Electrical and Computer Engineering Mississippi State, Mississippi, October 2000
- [5] C. Y. Suen, "N-gram Statistics for Natural Language Understanding and Text Processing," *IEEE Trans. on Pattern Analysis & Machine Intelligence*. PAMI, 1(2), pp.164-172, April 1979.
- [6] A. Zamora, "Automatic Detection and Correcting of Spelling Errors in A Large Data Base," *Journal of the American Society for Information Science*. 31, 51, 1980.
- [7] J. L. Peterson, "Computer Programs for Detecting and Correcting Spelling Errors," *Comm. ACM* 23, 676, 1980.
- [8] E. M. Zamora, J. J. Pollock, and Antonio Zamora, "The Use of Trigram Analysis for Spelling Error Detection," *Inf. Proc. Mgt.* 17, 305, 1981.
- [9] J. J. Hull and S. N. Srihari, "Experiments in Text Recognition with Binary N-gram and Viterbi Algorithms," *IEEE Trans. Pattern Analysis & Machine Intelligence*, PAMI-4, 520, 1980.
- [10] J. J. Pollock, "Spelling Error Detection and Correction by Computer: Some Notes and A Bibliography," *J. Doc.* 38, 282, 1982.
- [11] R. C. Angell, G. E. Freund, and P. Willette, "Automatic Spelling Correction Using Trigram Similarity Measure," *Inf. Proc. Mgt.* 18, 255, 1983.
- [12] E. J. Yannakoudakis, P. Goyal, and J. A. Huggill, "The Generation and Use of TextFragments for Data Compression," *Inf. Proc. Mgt.* 18, 15, 1982.
- [13] J. C. Schmitt, "Trigram-based Method of Language Identification," U.S. Patent No. 5,062,143, 1990.
- [14] W. B. Cavnar and J. M. Trenkle, "N-gram-based Text Categorization," *Proceeding of the Symposium on Document Analysis and Information Retrieval*, University of Nevada, Las Vegas, 1994.
- [15] P. Willett, "Document Retrieval Experiments Using Indexing Vocabularies of Varying Size. II. Hashing, Truncation. Digram and Trigram Encoding of Index Terms." *J. Doc.* 35, 296, 1979.
- [16] W. B. Cavnar, "N-gram-based Text Filtering for TREC-2," *The Second Text Retrieval Conference (TREC-2)*, NIST Special Publication 500-215, National Institute of Standards and Technology, Gaithersburg, Maryland, 1994.
- [17] Marc Damashek, "Gauging Similarity via N-grams: Language-independent Sorting, Categorization, and Retrieval of Text," *Science*, 267, pp.843-848, 1995.

[18] L. Bahl *LR*, Balakrishnan-aiyer s, Franz m, Gopalkrishnan ps, Gopinath r, Novak m, Padmanavan m, roukos s, "The IBM Large Vocabulary Continuous Speech Recognition System for the ARPA NAB News Task," Proc ARPA Workshop on Spoken Language Technology, pp. 121-126,1995.

[19] Johannes Furnkranz "A Study Using n-gram Features for Text Categorization "  
Austrian Research Institute for Artificial Intelligence Technical Report OEFAI-TR-98-30  
Schottengasse 3, A-1010 Wien, Austria

[20] Ethan Miller, Dan Shen, Junli Liu and Charles Nicholas "Performance and Scalability of a Large-Scale N-gram Based Information Retrieval System" Journal of Digital information, volume 1 issue 5.