Real Time Word Prediction Using N-Grams Model

Jaysidh Dumbali, Nagaraja Rao A.

***Abstract:** Predicting the most probable word for immediate selection is one of the most valuable technique for enhancing the communication experience. With growth in mobile technologies and vast spread of the internet, socializing has become much easier. People around the world spend more and more time on their mobile devices for email, social networking, banking and a variety of other activities. Due to fast paced nature of such conversation it is necessary to save as much as time possible while typing. Hence a predictive text application is necessary for this. Text prediction is one of the most commonly used techniques for increasing the rate of communication. However, the speed at which text is predicted is also very important in this case. The objective of this work is to design and implement a new word predictor algorithm that suggests words that are grammatically more appropriate, with a lower load for system and significantly reduce the amount of keystrokes required by users. The predictor uses a probabilistic language model based on the methodology of the N-Grams for text prediction.*

***Keywords:** Natural Language processing, N-Grams Stupid Backoff, Predictive text analytics, GoodTuring algorithm.*

I. INTRODUCTION

The ability of predicting the next word based upon its adjacent words is one of the fundamental task in natural language processing (NLP). In NLP the major focus is on understanding and determining how the interaction between human and a computer can be optimized. As humans, we tend to use language based upon the situation we are presented with. Selection of our next word depends upon a set of previous words. The goal of this research is to replicate a similar behavior of human word selection into a natural language processing model. Suggestions of new words that might be used are generated based upon the previous set of words. To achieve this goal, the N-Grams model is used. We assign probability to each word and select the next word with highest probability values.Thisprobability values are generated based upon the sequence of previous words. These sequences are known as N-Grams where N is the value which may be a unigram, bigram, trigram, quad gram.An important aspect which needs to be understood is the speed at which the words are predicted. Since one of the popular application of text prediction is in mobile communication, it is necessary that the load on the system as well as the speed of prediction should be optimal. This requires some tradeoff between the accuracy of the predicted words and speed. To achieve a modest speed, we limit the N-Gram model to quad grams. The data used for training the model are also cleaned to eliminate all the stop words and profanity word.

Manuscript published on 30 March 2019.
*****Correspondence Author(s)
Jaysidh Dumbali, Vellore Institute of Technology Vellore, Tamil Nadu, INDIA
Nagaraja Rao A. Vellore Institute of Technology Vellore, Tamil Nadu, INDIA

I. LITERATU REREVIEW

Predicting the next word has been an important technique for better communication for more than a decade. Traditional systems used word frequency lists to complete the words already spelled out by the user.

However, in the last few years more advanced predictive techniques have emerged based on the preceding word or syntactic rules. More advanced prediction methods can save keys to higher rates. Even though several researchers have found out that the increased cognitive load associated with word prediction may affect with the quick communication, recent findings have stated that more accurate predictions can compensate more as compared to these loads [1]. The benefits of increased accuracy of prediction precision cannot be confined to keystrokes saved by the predictions. An efficient word prediction model can improve the quality along with quantity of text generated for persons with language impairments, and those with learning disabilities [2]. Word prediction techniques can also be used in order to separate ambiguous key pad sequences, correct typing errors and provide more accurate scanning interface features forecasts [3]. The prediction of letter sequences was analyzed by Shannon (1951). He discovered that written English has high level of repetitions. Based upon this research an obvious question was whether users can be supported by systems which forecast the next keystrokes, words, or phrases while writing text [4]. A Unix shell was developed by Matodanad Yoshida [5] which predicts the command which a user is most likely to use based upon the current history of already entered commands. This has beenenchanced by Korvemaker and Greiner [6] which would predict the entire command lines. Similar study has also been done by Jacobs and Blockeel [7] who for preciting the next command using Markov models. A variety of typing tools for apraxiapersons and dyslexic persons were developed within natural language by Garay-Vitoria andAbascal, 2004; Zagler and Beck [8]and Magnuson and Hunnicutt [9]. These tools provide a possible list of words from which the user could select the required or most approximate word. For these users it is usually more efficient to scan and choose from lists of proposed words than to type. Scanning and selecting skilled authors from several displayed options may, in contrast, slow down (Langlais et al. 2002; Magnuson and Hunnicutt 2002). A prediction system can make more suitable word choices for the user by exploiting the present sentence context using statistical techniques. The previous n-1 word is used in predicting the current (nth) term in the N-Grams word prediction methods. In a large corpus, known as the training text, the N-Grams data is collected by counting each single n-word-sequence. In case of Augmented communications usage N-Grams techniques were limited to unigram and bigram word prediction, but in many other areas related to natural language processing such as speech recognition and machine translation trigram and higher N-Grams orders were often used [10].

There are such numbers of linguistically valid n word sequences for N-Grams-order orders higher than unigrams, that certain sequences do not appear or occur to provide statistically meaningful data even in extensive training texts. Consequently, a prediction system needs to temper its predictions of N-Grams higher order with lower, more reliable predictions of N-Grams. In general, the process of linear interpolation weighs predictions from each N-Grams order by another factor [11]. Eventually, the accuracy of a N-Grams prediction model heavily depends on the size of the text, even when using this method.

II. PROBLEM ONHAND

The main problem addressed in this paper is predicting the next appropriate word in a conversation. To achieve this, the model has been trained using various sources of data. The sources of data include various blogs, news and tweets, the detailed description of the same has been mentioned in section IV. The challenge is to predict the words with minimum waiting time.Theproblemonhand,therefore,isto lopamodel which is not just accurate but also fast in prediction.Theremainingpaperdescribes the approach used for this andalsoreportsontheperformanceofthealgorithmdeveloped in this way.

III. DATA DESCRIPTION

The data required for training and testing of the model are derived from three sources which includes blogs, news and twitter tweets. The combined size of the data is about 556 MB. The data consists of large chunks of text and the language used is English. The breakdown of the data sources is shown below in table 1.

Table 1: Data Summary Statistics

File Name	File Size	Line count	Word count
Blogs	200 MB	8,99,288	3,73,34,131
Tweets	160 MB	23,60,148	3,03,73,583
News	196 MB	10,10,243	3,43,72,529

IV. PROPOSEDMETHOD

The architecture of the proposed system can be divided into three section. The first section deals with cleaning the data sources. The second section deals with using the training corpus to generating the prediction tables by applying N-Grams model and the third section deals with using the test data to estimate the accuracy of the model. For training the model we first clean the data sets. Cleaning includes removing the stop words and profanity words. This cleaned data sets are now used for creation of Train and Test Corpora. The training and testing data sets are divided into 80:20 ratios. We apply ‘Good Turing’ algorithm for smoothing the training data. Smoothing is required for removing the irregularities and improving the accuracy of the model. Once smoothing is done N-Grams are generated from the Training Data Set. The maximum value of N is 4 thus generating Unigram, Bigrams, Trigrams and 4-Grams models.These models are then supplied to stupid back off algorithm. The basic concept behind stupid back off is that if a sequence is not present in a higher-order N-gram, we remove the first word and back off to a lower-order N-gram. The intermediate data obtained is then used for generating prediction tables. The prediction table is responsible for predicting the next words. It has been categorized to predict the next word based upon 1, 2 or 3-word sequence. This prediction generated are used for estimating the accuracy of the model. Each of these section is explained in depth below.

A. Cleaning the Data Sources

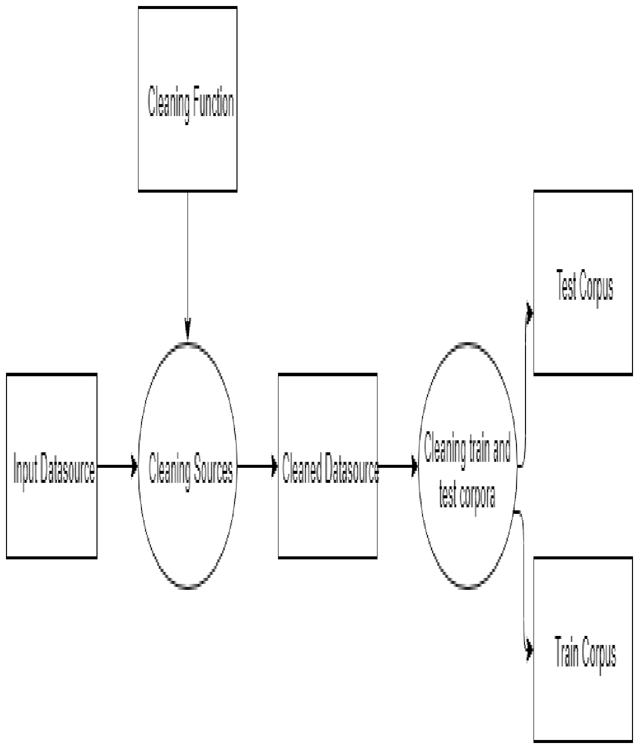

Figure 1: Cleaning of data sources and creating Test and Train corpus

Figure 1 shows the process of cleaning the data sources and generating the training and testing models. Cleaning function cleans a source of text (blogs, news, or twitter) using standard and ad-hoc basic cleaning functions. The standard cleaning functions came from R base and the 'tm' package. The result is a normalized version of the source, keeping its structure. Table 2 summaries the list of functions applied.

Table 2: List of cleaning functions

List of cleaning functions
Removing non-English text, Replacing links, Normalizing quotation, Removing numbers, Marking end of sentences, Removing punctuation, Remove non-Latin characters, Converting to lower case, Deleting profanities, Removing non-functional marks, Strip streams of <s/>, Normalizing white space

B. Generating Prediction Tables

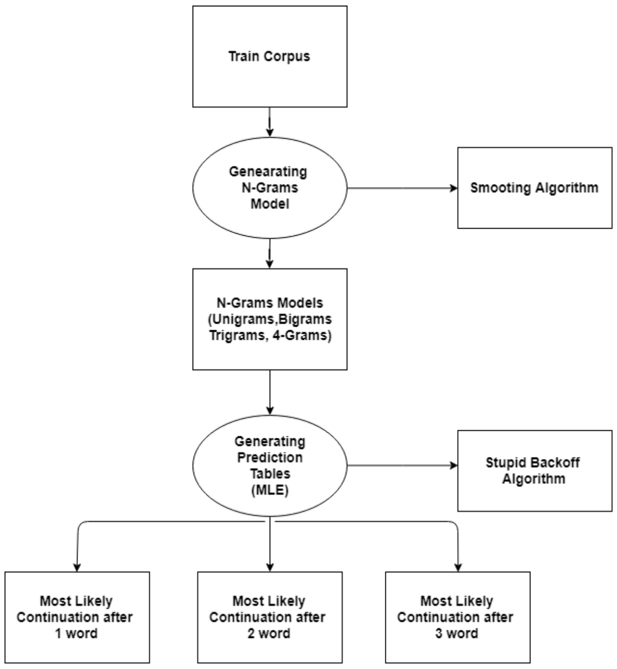
Train Corpus
Genearating N-Grams Model
Smooting Algorithm
N-Grams Models (Unigrams,Bigrams Trigrams, 4-Grams)
Generating Prediction Tables (MLE)
Stupid Backoff Algorithm
Most Likely Continuation after 1 word
Most Likely Continuation after 2 word
Most Likely Continuation after 3 word

Figure 2: Generating prediction tables by applying N-Grams model on the training corpus

Figure 2 depicts the process of generating the prediction tables.

The input to this module is the training corpora. We smooth the data before beginning the process of generating N-Grams. We use good Turing algorithm[11] for smoothing. Good Turning Algorithm is used for smoothing the data to remove noise. Let $N_r$be the count of n grams present in the training data, where r stands for count of occurrence. For example, {"a man", "a cat", "a man", "a fish", "rabbit ears", "a man"," rabbit ears"}, then $N_1 = 2$, $N_2 = 2$ and $N_3 = 1$. If N is the size of the training data, then the probability space occupied by n grams with rate r+1 is $(r+1)\,Nr/N$. For a n gram x that occurs r times, the probability of Good Turing is given by

$$P(x) = (r+1)\,\frac{(Nr+1)}{(N\cdot Nr)}$$

After smoothing, prediction tables are generated. The prediction table contains the probability of words and it helps to determine which words is most likely to appear next. It always looks for the best next words table. Stupid back off algorithm[12] is used for handling out of vocabulary sequences. In this algorithm if a sequence is not present in a higher-order N-grams, we remove the first word and back off to a lower-order N-gram, weighing the result by a fixed factor 'lambda' (in our case lambda = 0.4, as recommended by proponents of the algorithm). The result is a list of the n grams finally used and the list of most probable three next words for this n gram (in descending order). Table 3 shows the Tri-grams prediction table generated.

Table 3: Sample of trigram prediction table

Phrase	Word 1	Word 2	Word 3
perfect chemistry	Is	And	Between
perfect chocolate	Chip	And	Cake
perfect choice	For	To	Of
perfect Christmas	Gift	Tree	And

C. Predicting Next Word and Estimating Accuracy

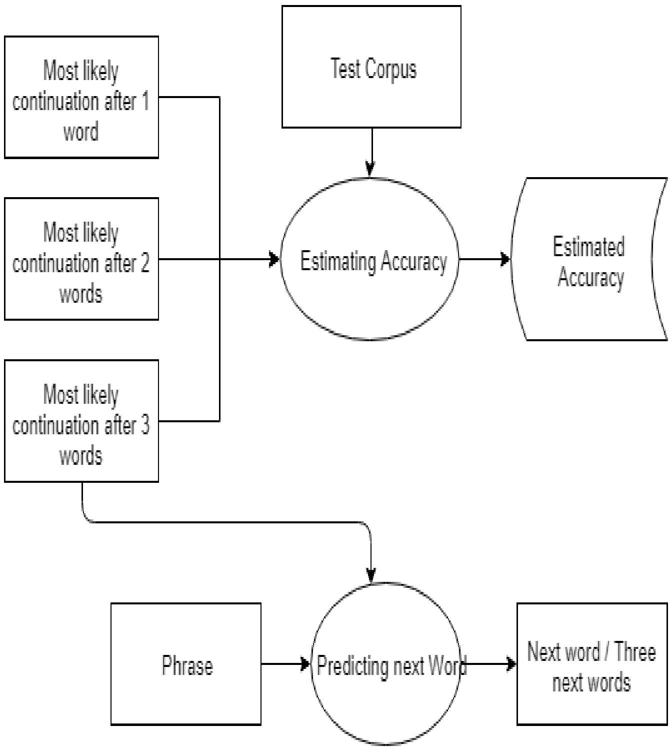
Most likely continuation after 1 word
Test Corpus
Most likely continuation after 2 words
Estimating Accuracy
Estimated Accuracy
Most likely continuation after 3 words
Phrase
Predicting next Word
Next word / Three next words

Figure 3: Predicting the next word and estimating the accuracy

This module is responsible for evaluating the performance of a n gram model against an unseen corpus of text. For this module we give input the testing dataset as an unseen corpus of text. Accuracy as a ratio of correct predictions over total predictions is estimated from each n gram order. Two options are evaluated: a prediction is right if it matches only the first next word predicted; or it is right if it is in the three next word proposed. The detailed analysis of the results is given in the next section.

IV. RESULT ANALYSIS

The simplest way to evaluate how our language model performs is measuring the accuracy: the ratio between number correct predicted words and total number of predictions. We have measured the accuracy of our model against a sample of 100,000 different sequences extracted from the test subset, which was not used in the training phase to avoid biased results. With the goal of reducing processing time and the loading cost and data footprint of the models in the server, we have curbed the N-grams applying three criteria:

a) Pruning the N-grams with total frequency below 3.

b) Covering word sequences that were not present in the training corpora ('out of vocabulary' OOV sequences) through the 'stupid backoff' algorithm: if a sequence is not present in a higher-order N-gram, we remove the first word and backoff to a lower-order

N-gram, weighing the result by a fixed factor 'lambda' (in our case lambda = 0.4, as recommended by proponents of the algorithm).

c) Reorganizing the N-grams selecting only the three most likely continuations for each sequence of words detected in the training corpora (after scoring results with stupid back off).

Table 4 summarizes the results using N-grams of increasing order (backed by the lower-order ones in case of OOV sequences) and considering as a correct prediction both cases: only the most likely word ('accuracy_1w'); or anyone of the three most likely ('accuracy_3w'):

Table 4: Accuracy of the models.

Model	**Accuracy 1 word**	**Accuracy 3 words**	**Time (seconds)**
Bigrams	11.57	20.46	95.85
Trigrams	14.9	24.97	126.81
4-grams	15.98	26.08	135.14

A sample of 2000 words were taken from the test dataset. The time was calculated as total time required by the N-Gram modelsto go through all the n-grams. The table 5 shows the hit (correct predictions) and the overall predictions estimated for each N-Grams model.

Table 5: Hits and total predictions of the models

Model	**Hit**	**Total**
Bigrams	4202	7873
Trigrams	6143	10682
4-Grams	7127	11737

Although accuracy figures seem modest at first glance considering the number of different words in the corpora vocabulary that potentially can be selected as next word (~675,000) and the relative simplicity of the N-grams models, they are quite remarkable.

Also the time required by each model is quite impressive consider the total number of predictions it needed to scan,

VII. CONCLUSION AND FUTURE SCOPE

This paper shows that N-Grams being a fairly simple language model can prove to be efficient for real time prediction requirements. The accuracy of the model can be seen improving as the value of N-Grams increases.Consider further improvements: building a 5-grams model; avoiding pruning when while training the models; or applying more powerful smoothing algorithms than 'stupid backoff' can be applied for increase the accuracy of the model. However, as the size of model increase the prediction time would also increase. Hence the tradeoff between accuracy and speed needs to be considered while enhancing the model.

REFERENCE

1. Horstmann Koester, H. & Levine, S. (1996). Effect of a word prediction feature on user performance. Augmentative and Alternative Communication, 12, 155-168.
2. Morris, C., Newell, A., Booth, L., Ricketts, I., & Arnott, J. (1992). Syntax PAL: A system to improve the written syntax of language-impaired users. *Assistive Technology*, 4, 51-59.
3. Lesher, G., Moulton, B., & Higginbotham, D. (1998). Techniques for augmenting scanning cmunication. *Augmentative and Alternative Communication*, 14, 81-101.
4. Shannon. 1951. Prediction and entropy of printed english. In *Bell Systems Technical Journal, 30, 50-64.*
5. Motoda and K. Yoshida. 1997. Machine learning techniques to make computers easier to use. In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence.
6. Korvemaker and R. Greiner. 2000. Predicting Unix command lines: adjusting to user patterns. In Proceedings of the National Conference on Artificial Intelligence.
7. Jacobs and H. Blockeel. 2001. The learning shell:automated macro induction. In Proceedings of the International Conference on User Modelling.
8. Garay-Vitoria and J. Abascal. 2004. A comparison of prediction techniques to enhance the communication of people with disabilities. In Proceedings of the 8th ERCIM Workshop User Interfaces For All.
9. Magnuson and S. Hunnicutt. 2002. Measuring the effectiveness of word prediction: The advantage of long term use. Technical Report TMH-QPSR Volume 43, Speech, Music and Hearing, KTH, Stockholm, Sweden.
10. Jelinek, F. (1990). Self-organized language modeling for speech recognition. In Waibel, A. & Lee, K. (Eds.), Readings in Speech Recognition, pp. 450-506. San Mateo, California: Kaufmann.
11. Orlitsky, A., Santhanam, N. P., & Zhang, J. (2003). Always good turing: Asymptotically optimal probability estimation. Science, 302(5644), 427-431.
12. T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean,“Large language models in machine translation,” in In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing (EMNLP) and Computational Natural Language Learning (CoNLL), 2007.

Jaysidh Dumbali,Nagaraja Rao A.
SCOPE, VIT VELLORE, INDIA
jay.dumbali@gmail.com,nagarajaraoa@vit.ac.in