International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 04 | April - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

WORD WISE PREDICTION USING DEEP LEARNING

Kakumanu Lakshmi Sireesha[1], Jetti Aakanksha[2], Kothapalli Nikita[3],

Mummadi Sesha Sai[4], S. Anil Kumar[5]

[1,2,3,4] Student, Department of Computer Science and Engineering, Tirumala Engineering College
[5] Professor, Department of Computer Science and Engineering, Tirumala Engineering College

---

Abstract - Word wise prediction is an input technology that simplifies the process of typing by suggesting the next word for a user to select, as typing in a conversation consumes time. You might be using it daily when you write texts or emails without realizing it.

Most of the keyboards in smart phones suggest next word prediction features. Google also uses next word prediction based on our browsing history. So, preloaded data is also stored in the keyboard function of our smart phones to predict the next word correctly.

By predicting the next word in a sequence, the number of keystrokes of the user can be reduced. In this project, we have used a deep learning approach by using Long Short Term Memory(LSTM) technique which gives better prediction accuracy than the machine learning approach.

***Key Words*:** Word, technology, smart phones, keystrokes, long short term memory.

1.INTRODUCTION

Word prediction tools were developed which can help to communicateand also to help the people with less typing speed. The research on word prediction has been performing well. Word prediction technique does the task of guessing the preceding word that is likely to continue with few initial text fragments. Existing systems work on a word prediction model, which suggests the next immediate word based on the current available word. These systems work using machine learning algorithms which have limitations to create accurate sentence structure. Developing technologies have been producing more accurate outcomes than the existing system technologies, models developed using deep learning concepts are capable of handling more data efficiently.

This is similar to how a predictive textkeyboard works on apps like What's App, Facebook Messenger, Instagram, e-mails, or even Google searches. It will consider the last word of a particular sentence and predict the next possible word. You might be using it daily when you write texts or emails without realizing it. Most of the keyboards in smartphones give next word prediction features; google also uses next word prediction based on our browsing history. So, preloaded data is also stored in the keyboard function of our smartphones to predict the next word correctly. By predicting the next word in a sequence, the number of keystrokes of the user can be reduced. Three deep learning techniques namely Long Short Term Memory (LSTM), Bi-LSTM and BERT have been explored for the task of predicting the next word. In this project we have used the LSTM technique for predicting the next word.

2. LITERATURE REVIEW

- **S. Lai, et. al. [1]** have proposed the context based information classification; RCNN is very useful. The performance is best in several datasets, particularly on document-level datasets. Depending on the words used in the sentences, weights are assigned to it and are pooled into minimum, average and the max pools. Here, max poolingis applied to extract the keywords from the sentences which are most important. RNN, CNN and RCNN when compared with other traditional methods such as LDA, Tree Kernel and logistic regression generate highly accurate results.

- **Hassan, et. al. [9]** have proposed RNN for the structure sentence representation. This tree like structure captures the semantics of the sentences. The text is analyzed word by word by using RNN then the semantics of all the previous texts are preserved in a fixed size hidden layer. For the proposed system LSTM plays an important role, being a memory storage, it holds the characters which helps in predicting the next word.

- **J. Y. Lee, et. al. [7]** have proposed that text classification is an important task in natural language processing. Many approaches have been developed for classification such as SVM (Support Vector Machine), Naive Bayes and so on. Usually short text appears in sequence (sentences in the document) hence using information from preceding text may improve the classification.

- **Z. Shi, et. al. [4]** have defined that recurrent neural network have input, output and hidden layers. The current hidden layer is calculated by the current input layer and previous hidden layer. LSTM is a special Recurrent Neural Network. The repeating module of ordinary RNN has a simple structure; instead, LSTM uses a more complex function to replace it for more accurate results. The key element in the LSTM is the cell state which is also called the hiddenlayer state.

- **J. Shin, et. al. [10]** have defined that understanding the contextual aspects of a sentence is very important and reveals significant contributions in the field of natural language processing (NLP) and deep learning. Their work likely explores advancements.

3. METHODILOGY

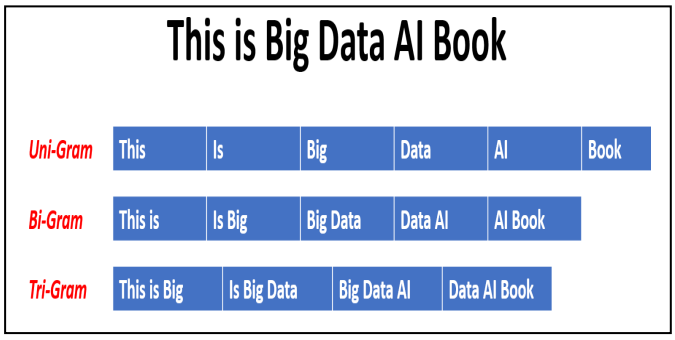
3.1 EXISTING SYSTEM

An n-gram model is a type of probabilistic language model for predicting the next item in such a sequence in the form of a (n — l)— order Markov model. The N-gram model predicts the occurrence of a word based on the occurrence of its N — 1 previous words.

You can think of an N-gram as the sequence of N words, by that notion, a 2-gram (or bigram) is a two-word sequence of words like "please turn", "turn your", or" your homework", and a 3-gram (or trigram) is a three-word sequence of words like "please turn your", or'turn your homework'.

Disadvantages

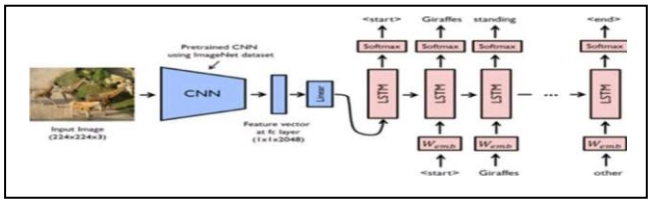
- Markov chains don't have memory as we are suggesting the next word basedon frequency.
- Sparsity problem occurs with increasing values of n.
- Storage problems occurs due to large number of n-grams from largevocabulary size.
- N-grams only consider a fixed number of preceding words (N) to predict thenext word.



.3.2 PROPOSED SYSTEM

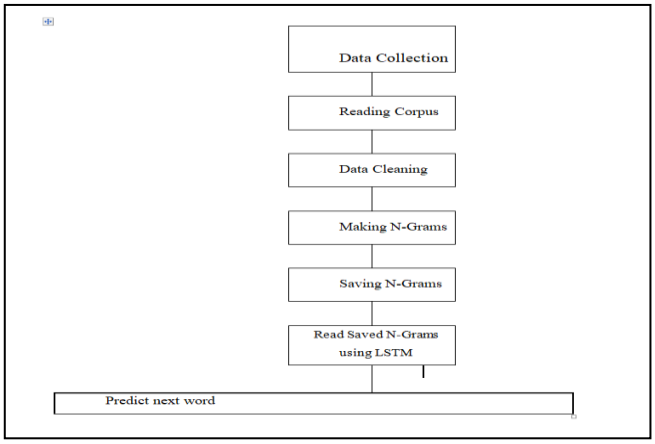
RNNs suffer from the vanishing gradient problem which makes it difficult tolearn and tune the parameters in the earlier layers. In the proposed system, long short-term memory (LSTM) networks were later introduced to overcome this limitation.

An LSTM, Long Short-Term Memory, model was first introduced in the late 90s by Hoch Reiter and Schmid Huber. Since then, many advancements have been made using LSTM models and its applications are seen from areas including time series analysis to connected handwriting recognition. An LSTM network is a type of RNN which learns dependence on historic data for a sequence prediction task. What allows LSTMs to learn these historic dependencies are its feedback connections. For a common LSTM cell, we usually see an input gate, an output gate, and a forget gate. The weights of these gates control the flow of information in an LSTM model and thus are the parameters learnt during the training process.

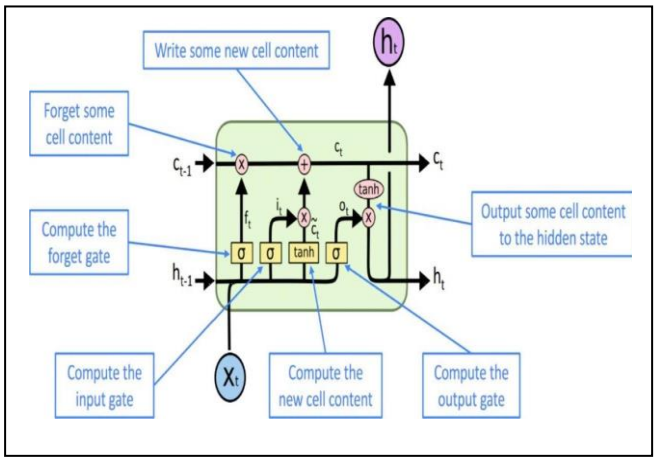


Phases of Proposed Method

A corpus is a collection of authentic text or audio organized into datasets. In natural Language processing, a corpus contains text and speech data that can be used to train Aland machine learning systems.



Block diagram of Next Word Predictor Using LSTM



PROPOSED ALGORITHM

Algorithm: Next word Prediction Using LSTM
Input: Text Corpus data.
For 1 to P // LSTM classifiers

For 1 to N // suggesting unigram or Bigram or Trigram

1.Generating the Subset n

2.Generate N-grams instances from the whole training set.

3.Randomly choose n.

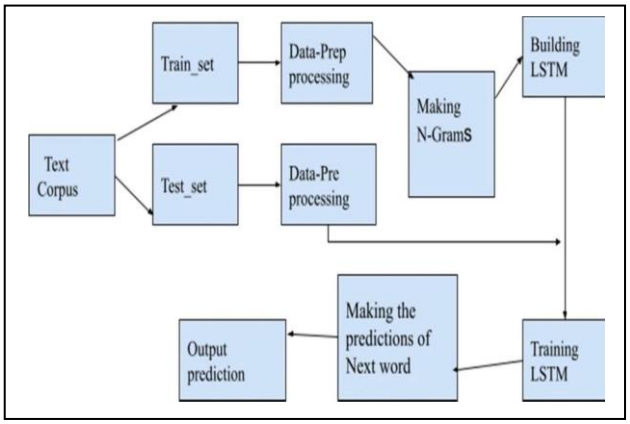
4.Training the individual

5.LSTM classifier Train up to p^{th} classifier.

6.Making a prediction For the given Input
7.Predict the outcome withNext word End

8.Predict the Next word Based on N-gram.

SYSTEM FRAMEWORK



DATA PROCESSING

Exploratory analysis

The input files had garbage text such as repeated letter words (e.g., "aaaa", "qqqqqqqq"). The single-word, Document Term Matrix (DTM)showed a high number of sparse terms. Although the texts are mostly in English, they contain words from otherlanguages.

Understanding word frequencies

It was not possible to produce a DTM for each n-gram because it kept reporting errors. For this reason, discovering bigram and trigram frequencies was also not possible. After some research, as well as trial and error attempts that were consuming too much time, the strategy was changed to create each n-gram with repetitions removed, and creating a term frequency vector for all individual words.

Basic N- gram model Patterns

- FILTER PATTERN_OI: characters not matching "a" through "z" (lower anduppercase), digits ranging from 0 to 9, the blank space, and the single quote character (').
- FILTER PATTERN_02: text pattern starting with a single quote, followed by any number of letters ranging from "a" through "z" (lower and uppercase), and then followed by a blank space.
- FILTER PATTERN 03: text pattern matching truncated forms of the verbs 'todo", "to be", and "to have".
- FILTER PATTERN 04: "stand-alone" numbers, such as 2000.
- SINGLE_CHARACTER PATTERN: terms composed of one character,excluding a , i ”,and I”.

PROCESS

Words are converted to lowercase. Although capitalized proper nameswill be lost, there are many more portions of the texts that contain unnecessary capitalized words.

All characters not forming part of Fiter_Pattern_1 are removed. The blank space is included in the pattern just so that it is ignored. The single quote, usually located in between words (e.g., "don't" and "they've") is left in the texts. This way, these words can be processed later, that is, once the rest of the non-pattern characters have been removed.

4.IMPLEMANTATION

Keras

Keras is a powerful and ease-to-use free open source Python library for developing and evaluating deep learning models. It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code.

Keras was created to be user friendly, modular, easy to extend, and to work with Python. The API was "designed for human beings, not machines," and "follows best practices for reducing cognitive load."

Neural layers, cost functions, optimizers, initialization schemes, activation functions, and regularization schemes are all standalone modules that you can combine to create new models. New modules are simple to add, as new classes and functions. Models are defined in Python code, not separate model configuration files.

Google Colab

If you have used Jupyter notebook previously, you would quickly learn to use Google Colab. To be precise, Colab is a free Jupyter notebook environment that runs entirely in the cloud. Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs. Colab supports many popular machine learning libraries which can be easily loaded in your notebook.

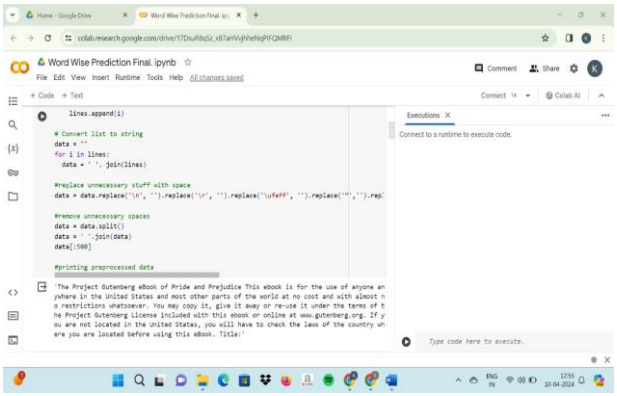
Python

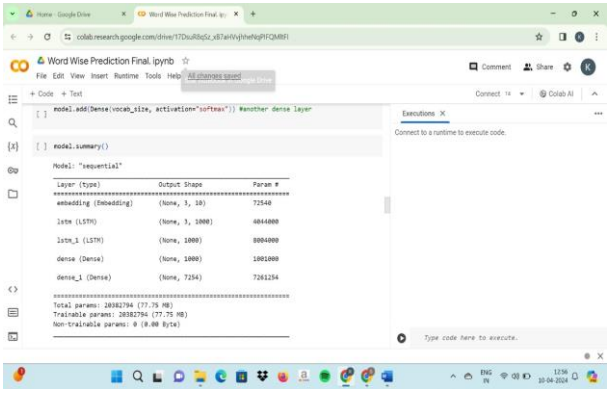
Python is a high level general purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. It contains various libraries which are useful to implement the logic like NumPy, Keras, OpenCV, etc.. It provides easy compilation and provides efficient output.

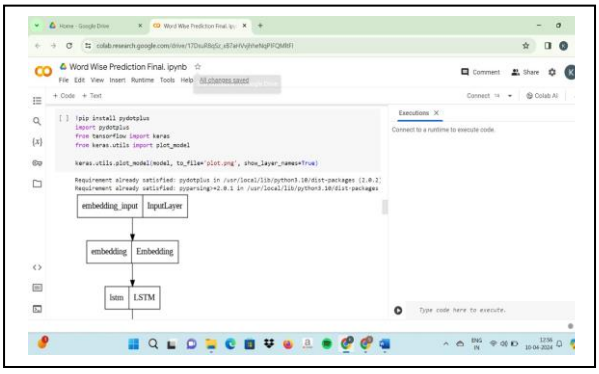
TensorFlow

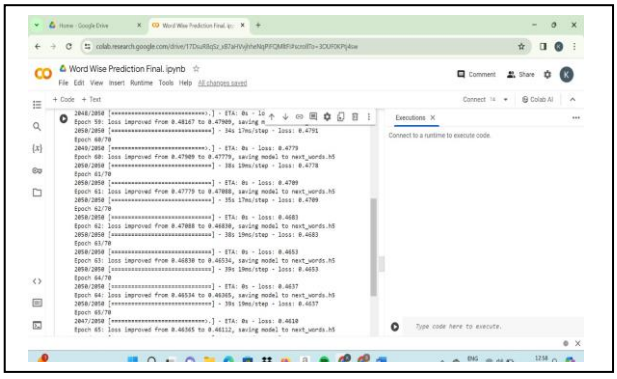
TensorFlow stands as a leading open-source machine learning library, developed by Google Brain. It boasts a robust ecosystem tailored to streamline the development and deployment of machine learning models, with a particular focus on neural networks. TensorFlow's strength lies in its flexible architecture, which supports diverse hardware platforms such as CPUs, GPUs, and TPUs (Tensor Processing Units), making it adaptable to a wide array of computational environments, from desktops to large-scale distributed systems.

5.RESULTS









6.TESTING

Software testing is a process used to help identify the correctness, completeness and quality of developed computer software. Software testing is the process used to measure the quality of developed software. Testing is the process of executing a program with the intent of finding errors. Software testing is often referred to as verification & validation.
STLC (Software Testing Life Cycle):

Testing itself has many phases i.e., is called as STLC. STLC is part of SDLC

- Test Plan
- Test Development
- Test Execution
- Analyze Result
- Defect Tracking

TYPES OF TESTING

1.White Box Testing
2.Black Box Testing
3.Grey Box Testing

WHITE BOX TESTING

White box testing as the name suggests gives the internal view of the software. This type of testing is also known as structural testing or glass box testing as well, as the interest lies in what lies inside the box.

BLACK BOX TESTING

It is also called as behavioral testing. It focuses on the functional requirements of the software. Testing either functional or nonfunctional without reference to the internal structure of the component or system iscalled black box testing

GREY BOX TESTING

Grey Box testing is a software testing method to test the software application with partial knowledge of the internal working structure. It is a combination of black box and white box testing because it involves access to internal coding to design test cases as white box testing and testing practices are done at functionality level as black box testing.

7.CONCLUSION

Word wise prediction powered by deep learning stands as a transformative technology, offering remarkable accuracy and fluency in anticipating the next word in a sequence of text. Through sophisticated algorithms and extensive training on vast datasets, deep learning models capture complex patterns and dependencies in language, enhancing user experience and productivity across various domains.

Furthermore, the continuous evolution of deep learning techniques and the availability of large-scale datasets have contributed to the ongoing improvement of next word prediction systems. Techniques such as transfer learning, fine-tuning, and ensemble methods enable models to leverage knowledge from pre-trained models and adapt to specific tasks or domains with minimal data requirements.
Additionally, the integration of attention mechanisms and contextual embeddings further enhances the models' ability to understand and generate contextually relevant predictions, leading to more precise and contextually coherent outcomes.

As deep learning continues to advance and researchers explore novel approaches to modeling language, the future holds great promise for further improvements in word wise prediction systems, ultimately shaping the way we interact with and leverage textual information in various applications and domains.

8.REFERENCES

1. R. Kneser and H. Ney, "Improved backing-off for n-gram language modeling", International Conference on Acoustics Speech and SignalProcessing, pp. 181-184, 1995.

2. S.F. Chen and J.T. Goodman, "An empirical study of smoothing techniques for language modeling", Computer Speech and Language,vol. 13, no. 4, pp. 359-393, 1999.

3. J. Goodman, "A bit of progress in language modeling", MicrosoftResearch, 2001.
4. Yoshua Bengio, Rejean Ducharme, Pascal Vincent and Christian Jauvin,"A neural probabilistic language model", Journal of Machine Learning Research, vol.

5. J. Allen, Natural Language Understanding, Benjamin/CummingsPublishing, 1995.

6. Fu-Lian Yin, Xing-Yi Pan, Xiao-Wei Liu and Hui-Xin Liu, Deep neuralnetwork language model research and application overview, 2015.

7. Miltiadis Allamanis, Earl T Barr, Premkumar Devanbu and Charles Sutton, A survey of machine learning for big code and naturalness, 2017.

8. Mohd. Majid and Piyush Kumar, Language Modelling: Next wordPrediction, 2019.
9. "The list of books "Beyond Good and Evil by Friedrich WilhelmNietzsche" from the official Library" in The Project Gutenberg.

10. The list of Free eBooks - Project Gutenberg books" from the officialLibrary' The Project Gutenberg.