**ABSTRACT**

The primary goal of **“NEXT WORD PREDICTION WITH NLP AND DEEP LEARNING”** is to implement a deep learning based system to predict next word(s) for given a set of current words to the user. The project proposes to study and implement the Long Short Term Memory network (LSTM) model to predict the next word in sentence. It proposes to use the Project Gutenberg’s Adventures of Sherlock Holmes, by A. Conan Doyle EBook text dataset for modeling and Python as programming language for implementation.

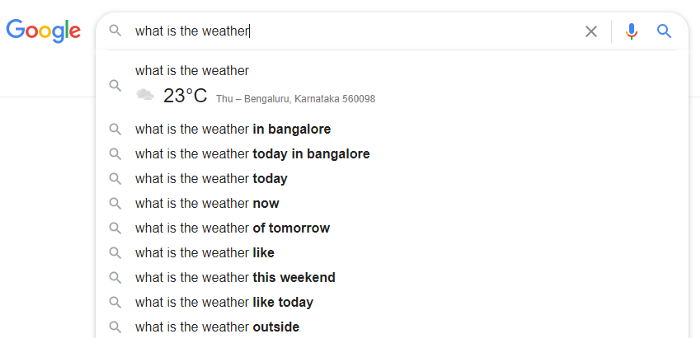
**CHAPTER I**

**INTRODUCTION**

* 1. **Introduction**

The main objective of this major project entitled “**NEXT WORD PREDICTION WITH NLP AND DEEP LEARNING**” is used predict the next word for given sentence. Next Word Prediction is also called Language Modeling. It is one of the fundamental tasks of NLP. Natural language processing has been an area of research and used widely in different applications. Next word prediction is the task of predicting which word comes next. It is always helpful when a suggestion pops up trying to predict the next word to be written. As writing an essay and framing a big paragraph are time-consuming it will help end-users to frame important parts of the paragraph and help users to focus on the topic instead of wasting time on what to type next [1]-[3].

The general next word prediction systems, first analyzes the data and pre-processes the data. Then the data is tokenized and finally subjected to build the deep learning model. The deep learning model are usually built using LSTM’s. As LSTM is Long short time memory, it will understand the past text and predict the words which may be helpful for the user to frame sentences and this technique uses a letter to letter prediction means it predicts a character to create a word [2].



**Figure 1.1 Next word(s) Prediction in Google [4]**

* + 1. **Importance of next word prediction.** The following some importance’s of word prediction [5]:
  + It is used directly in a variety of industries including tech, finance, healthcare, transportation, legal, military and government.
  + Most people have interacted with a language model in some way at some point in the day, whether it be through Google search, an auto complete text function or engaging with a voice assistant.
  + word prediction can speed up the writing process
  + support to confirm word selection
  + improve overall quantity and quality of text including sentence structure and grammatical accuracy of text
    1. **Application of Next word prediction.** The following are some applications of next word prediction [5]:
  + Google search
  + Spelling error detection are correction
  + speech recognition
  + machine translation
  + part-of-speech tagging
  + parsing
  + handwriting recognition
  1. **Project Definition**

The project “**NEXT WORD PREDICTION WITH NLP AND DEEP LEARNING**”, aims to implement Long Short Term algorithm to predict the next word for a sentence. Programming is proposed to be carried out in python language. Here prediction is done by using LSTM algorithm which is embedded in python library as packages like tensorflow and function.

* 1. **Objectives**

The objectives of the proposed major project works are:

* + Understand the text
  + Cleaning the text
  + Understand NLP and tensorflow packages
  + Tokenize text using NLP libraries
  + Build a model using LSTM algorithm
  + Train the model
  + Testing the model
  1. **Chapter organization**

The report of the major project is organized to have five chapters as follows:

CHAPTER I – Gives the introduction for next word prediction.

CHAPTER II – Gives the literature survey.

CHAPTER III – Performing Deep learning model LSTM algorithm on text data using python

CHAPTER IV – Gives the result and discussion.

CHAPTER V – Concludes the report.

**CHAPTER II**

**LITERATURE SURVEY ON NEXT WORD PREDICTION**

1. **J. Yang, H. Wang and K. Guo, "Natural Language Word Prediction Model Based on Multi-Window Convolution and Residual Network," in IEEE Access, vol. 8, pp. 188036-188043, 2020, doi: 10.1109/ACCESS.2020.3031200. [6]**

**Description**

Multi-window convolution(MRNN) algorithm is implemented, also they have created residual-connected minimal gated unit(MGU) which is short version of LSTM in this cnn try to skip few layers while training result in less training time and they have good accuracy by far using multiple layers of neural networks can cause latency for predicting n numbers of words[6] .

1. **K. Terada and Y. Watanobe, "Code Completion for Programming Education based on Recurrent Neural Network," 2019 IEEE 11th International Workshop on Computational Intelligence and Applications (IWCIA), Hiroshima, Japan, 2019, pp. 109-114, doi: 10.1109/IWCIA47330.2019.8955090. [7]**

**Description**

This paper used RNN algorithm and also they have used GRU another form of RNN for code completion problem as RNN help to predict next code syntax for users. Authors claim that their method is more accurate compare to existing methods. They have separated next word prediction in two components: within-vocabulary words and identifier prediction. They have used LSTM neural language model to predict within vocabulary words. A pointer network model is proposed to identifier prediction [7].

1. **Habib, Md AL-Mamun, Abdullah Rahman, Md Siddiquee, Shah Ahmed, Farruk. (2018). An Exploratory Approach to Find a Novel Metric Based Optimum Language Model for Automatic Bangla Word Prediction. International Journal of Intelligent Systems and Applications. 2. 47-54. 10.5815/ijisa.2018.02.05. [8]**

**Description**

Authors worked on Bangla Language. They have proposed a novel method for word prediction and word completion. They have proposed N-gram based language model which predicts set of words. They have achieved satisfactory results [8].

**4. Joel Stremmel, Arjun Singh. (2020). Pretraining Federated Text Models for Next Word Prediction using GPT2 [9]**

**Description**

As GPT is quite huge and costly model for this kind of task as word prediction is a simple project, GPT will only act like wasting useful resources for simple task. So GPT2 is used in this paper for word prediction [9].

**5. Minghui Wang, Wenquan Liu and Yixion Zhong, "Simple recurrent network for Chinese word prediction," Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan), Nagoya, Japan, 1993, pp. 263-266 vol.1, doi: 10.1109/IJCNN.1993.713907 [10]**

**Description**

This was first approach to tackle this kind of problem, the paper discusses about LM and perplexity algorithm which is base of natural language processing, This helps to make a 3D input data for modeling [10].

**6. M. K. Sharma, S. Sarcar, P. K. Saha and D. Samanta, "Visual clue: An approach to predict and highlight next character," 2012 4th International Conference on Intelligent Human Computer Interaction (IHCI), Kharagpur, 2012, pp. 1-7, doi: 10.1109/IHCI.2012.6481820. [11]**

**Description**

They used Predicting Next Character Highlighter (PNCH) for Indian language. It was more of text correction and less about next word prediction but was quite good to understand. The Method called hit and miss but accuracy is less and the model was not efficient for this kind of problem statement [11].

**7. S. M. Sarwar and Abdullah-Al-Mamun, "Next word prediction for phonetic typing by grouping language models," 2016 2nd International Conference on Information Management (ICIM), London, 2016, pp. 73-76, doi: 10.1109/INFOMAN.2016.7477536. [12]**

**Description**

In this paper they created a auto-next-keyword for Bengali language which was challenging and it was found out that it is hard to get good accuracy by using RNN algorithm as due to its vanishing gradients and heavy recurrent NN take more time to train and test [12].

**8. Partha Pratim Barman, Abhijit Boruah,A RNN based Approach for next word prediction in Assamese Phonetic Transcription,Procedia Computer Science,Volume 143,2018,Pages 117-123,ISSN 1877- 0509,https://doi.org/10.1016/j.procs.2018.10.359. [13]**

**Description**

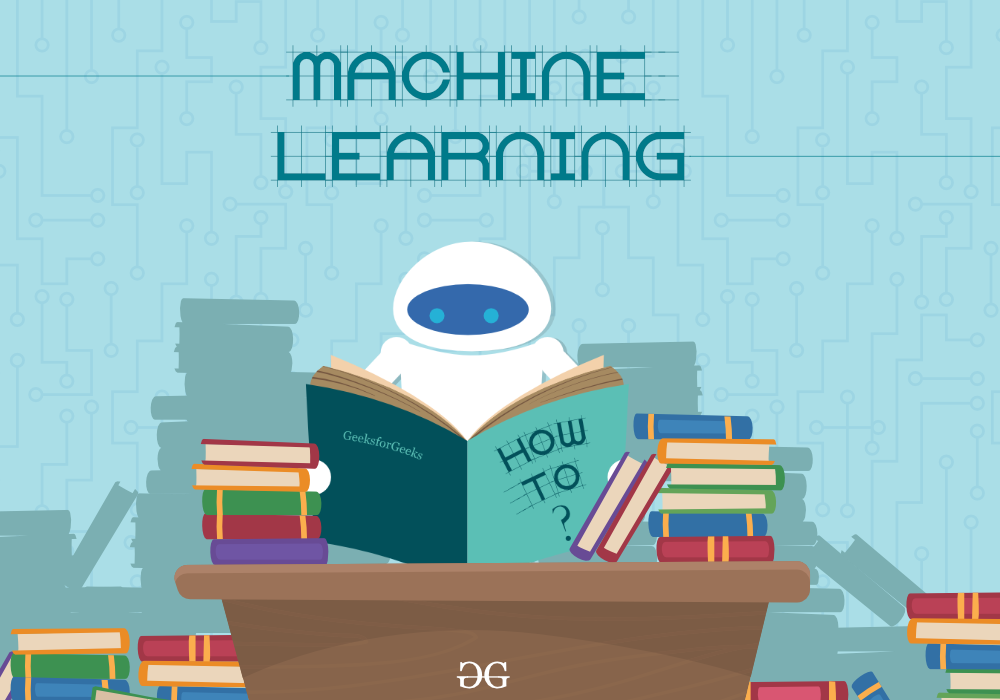
Authors have used LSTM for next word prediction for Assamese language. They have stored transcripted language according to International Phonetic Association (IPA) chart and fed to their model. They created a model for physically challenged people. This model uses Unigram, Bigram, Trigram, based Approach for next word prediction and was found out average to predict the word but accuracy was around 30-40 percentage [13].

**CHAPTER III**

**DESIGN OF THE PROPOSED SYSTEM FOR PREDICTING NEXT WORD WITH NLP AND DEEP LEARNING**

* 1. **Machine Learning**

Machine learning is a subset of artificial intelligence (AI) that provides system the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves. The process of learning beings with observation or data, such as examples, direct experience, or instruction, in order to look for pattern in data and make better decisions in the future based on the examples provided. The primary aim is to allow computers learn automatically without human intervention or assistance and adjust actions accordingly [14].



**Figure 3.1 Machine learning [15]**

# 3.1.1 Types of Machine Learning. Based on the methods and way of learning, machine learning is divided into mainly four types, which are:

1. Supervised Machine Learning
2. Unsupervised Machine Learning
3. Semi-Supervised Machine Learning
4. Reinforcement Learning

# Machine learning

# Figure 3.2 Types of machine learning [16]

# 3.1.1.1 Supervised Machine Learning. The computer is presented with example inputs and their desired outputs, given by a “teacher”, and the goal is to learn a general rule that maps inputs to outputs [17].

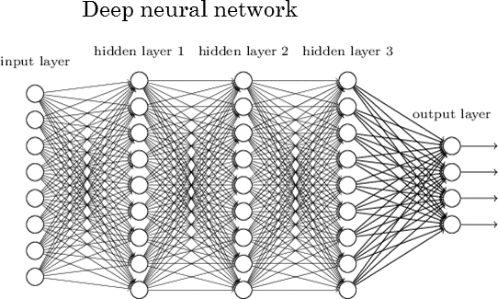
**3.1.1.2 Unsupervised Machine Learning.** No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a mean towards an end (feature learning) [17].

**3.1.1.3 Semi-Supervised Machine Learning.** Semi-supervised machine learning is a combination of supervised and unsupervised learning. It uses a small amount of labeled data and a large amount of unlabeled data, which provides the benefits of both unsupervised and supervised learning while avoiding the challenges of finding a large amount of labeled data [18].

**3.1.1.4 Reinforcement Learning.** A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). As it navigates its problem space, the program is provided feedback that’s analogous to rewards, which it tries to maximize [17].

* 1. **Deep Learning**

Deep learning is a subset of [machine learning](https://www.ibm.com/cloud/learn/machine-learning), which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain. It allowing to “learn” from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy [19].



**Figure 3.3 Deep Learning [20]**

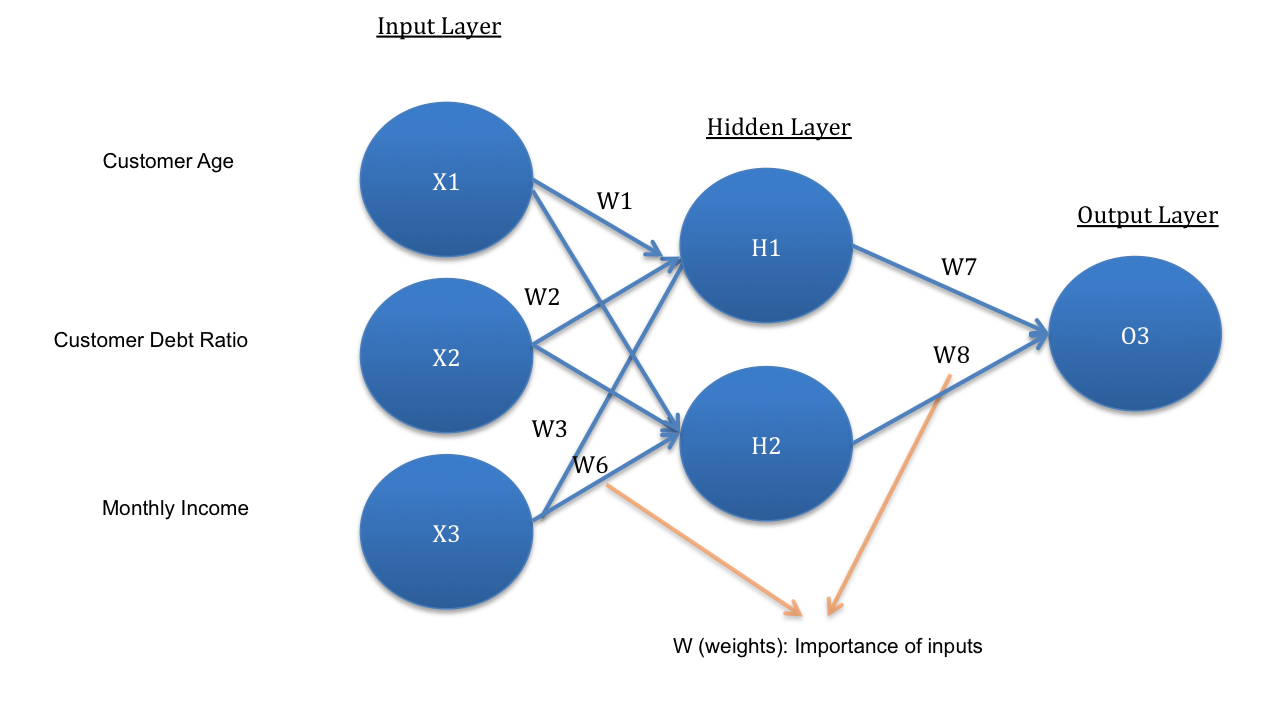
A Neural Network is made of an assortment of algorithms that are modeled on the human brain. These algorithms can interpret sensory data via machine learning, perception and label or the raw data. They are designed to recognize numerical patterns that are contained in vectors within which all the real-world data (images, sound, text, time, series, etc.) has to be translated [19].

Now we see three important types of neural network that from the basis for most pre-trained models in deep learning:

* Artificial Neural Network (ANN)
* Convolution Neural Network (CNN)
* Recurrent Neural Network (RNN)

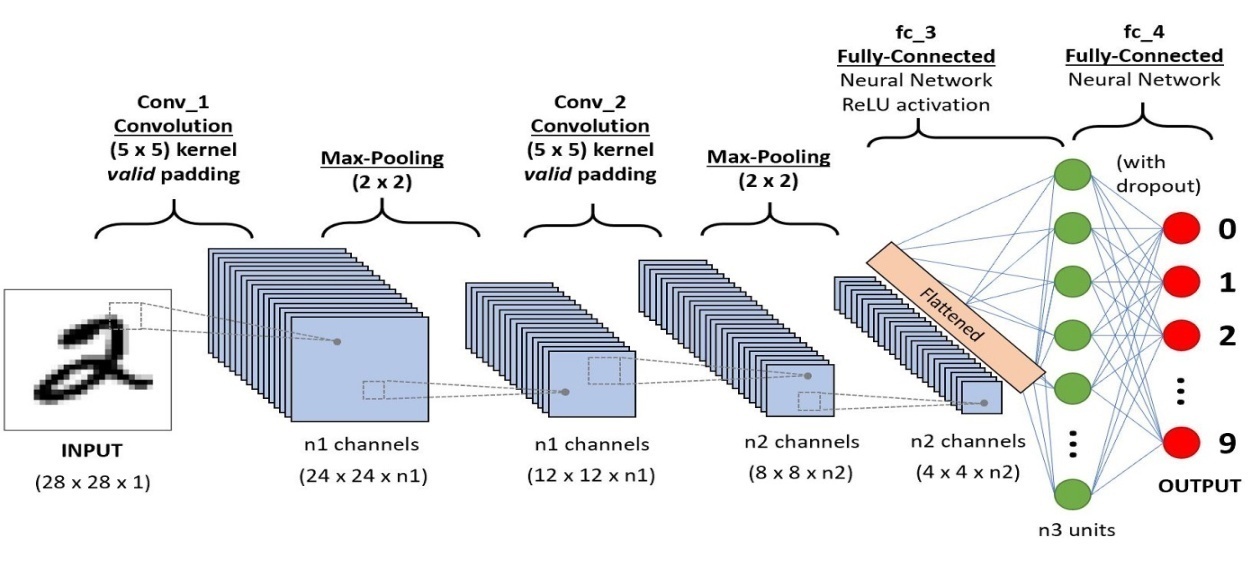
**3.2.1 Artificial Neural Network (ANN).** [Deep learning](https://www.techopedia.com/definition/30325/deep-learning) ANNs play an important role in machine learning ([ML](https://www.techopedia.com/definition/8181/machine-learning-ml)) and support the broader field of artificial intelligence ([AI](https://www.techopedia.com/definition/190/artificial-intelligence-ai)) technology. An artificial neuron network (neural network) is a computational model that mimics the way nerve cells work in the human brain. Artificial neural networks (ANNs) use [learning algorithms](https://www.techopedia.com/definition/33426/learning-algorithm) that can independently make adjustments - or learn, in a sense - as they receive new input. This makes them a very effective tool for non-linear statistical [data modeling](https://www.techopedia.com/definition/14/data-modeling) [21].

An ANN has hundreds or thousands of artificial neurons called processing units, which are interconnected by nodes. These processing units are made up of input and output units. The input units receive various forms and structures of information based on an internal weighting system and the neural network attempts to learn about the information presented to produce one output report. Just like human need rules and guidelines to come up with a result or output, ANNs also use a set of learning rules called backpropagation, an abbreviation for backward propagation of error, to perfect their output result [21].



**Figure 3.4 Artificial Neural Network (ANN) [22]**

**3.2.2 Convolution Neural Network (CNN).** A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics [22].



**Figure 3.5 Convolution Neural Network (CNN) [22]**

**3.2.3 Recurrent Neural Network (RNN).** RNNs are a powerful and robust type of neural network, and belong to the most promising algorithms in use because it is the only one with an internal memory.

Like many other deep learning algorithms, recurrent neural networks are relatively old. They were initially created in the 1980’s, but only in recent years have we seen their true potential. An increase in computational power along with the massive amounts of data that has to be worked with, and the invention of long short-term memory (LSTM) in the 1990s, has really brought RNNs to the foreground [23].

Because of their internal memory, RNN’s can remember important things about the input they received, which allows them to be very precise in predicting what is coming next. This is why they're the preferred algorithm for sequential data like [time series](https://builtin.com/data-science/introduction-segmentation-correlation-time-series-modeling), speech, text, financial data, audio, video, weather and much more. Recurrent neural networks can form a much deeper understanding of a sequence and its context compared to other algorithms. [23]

## 3.2.3.1 Long Short-Term Memory (LSTM). Long short-term memory (LSTM) is an [artificial neural network](https://en.wikipedia.org/wiki/Artificial_neural_network) used in the fields of [artificial intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence) and [deep learning](https://en.wikipedia.org/wiki/Deep_learning). Unlike standard [feedforward neural networks](https://en.wikipedia.org/wiki/Feedforward_neural_network), LSTM has feedback connections. Such a [recurrent neural network](https://en.wikipedia.org/wiki/Recurrent_neural_network) can process not only single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected [handwriting recognition](https://en.wikipedia.org/wiki/Handwriting_recognition), [speech recognition](https://en.wikipedia.org/wiki/Speech_recognition), [machine translation](https://en.wikipedia.org/wiki/Machine_translation), robot control, video games, and healthcare. LSTM has become the most cited neural network of the 20th century [24].

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. LSTM networks are well-suited to [classifying](https://en.wikipedia.org/wiki/Classification_in_machine_learning), [processing](https://en.wikipedia.org/wiki/Computer_data_processing) and [making predictions](https://en.wikipedia.org/wiki/Predict) based on [time series](https://en.wikipedia.org/wiki/Time_series) data, since there can be lags of unknown duration between important events in a time series, LSTMs were developed to deal with the [vanishing gradient problem](https://en.wikipedia.org/wiki/Vanishing_gradient_problem) that can be encountered when training traditional RNNs [24].

**3.2.3 Application of deep learning** Real-world deep learning applications are a part of ourdaily lives, some of application are [25]:

**Healthcare.** The healthcare industry has benefited greatly from deep learning capabilities ever since the digitization of hospital records and images. Image recognition applications can support medical imaging specialists and radiologists, helping them analyze and assess more images in less time [25].

**Law enforcement.** Deep learning algorithms can analyze and learn from transactional data to identify dangerous patterns that indicate possible fraudulent or criminal activity. Speech recognition, computer vision, and other deep learning applications can improve the efficiency and effectiveness of investigative analysis by extracting patterns and evidence from sound and video recordings, images, and a document, which helps law enforcement analyze large amounts of data more quickly and accurately [25].

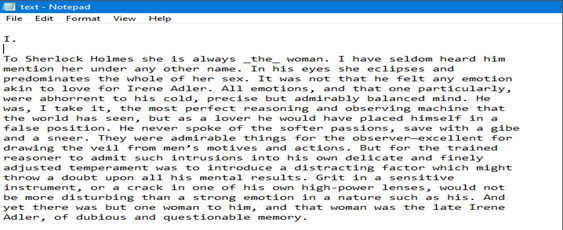
### Financial Services. Financial institutions regularly use predictive analytics to drive algorithmic trading of stocks, assess business risks for loan approvals, detect fraud, and help manage credit and investment portfolios for clients [25].

### Customer service. Many organizations incorporate deep learning technology into their customer service processes. Chatbots used in a variety of application, services, and customer service portals are a straightforward from of AI. Traditional Chatbots use natural language and even visual recognition, commonly found in call center like menus. However, more sophisticated chatbot solution attempt to determine, through learning if there are multiple responses to ambiguous question. Based on the response it receives, the chatbot then tries to answer these questions directly or route the conversation to a human user.Virtual assistants like Apple’s Siri, Amazon Alexa, or Google Assistant extends the idea of a chatbot by enabling speech recognition functionality. This creates a new method to engage to users in a personalized way [25].

* 1. **The Lifecycle of the proposed Deep Learning System**

The study and implementation of the next word prediction system is made by referring the article in reference [26].

**3.3.1 Discovery.** This main project proposes to study and implement design of a system for prediction of next word in sentence using LSTM algorithm.

**3.3.2 Data Collection.** Project Gutenberg’s Adventures of Sherlock Holmes, by A. Conan Doyle EBook used as text data in this project. The text data contains 581888 words. The section of the dataset shown in the figure below.

**Figure 3.6 Next word(s) Prediction dataset [27]**

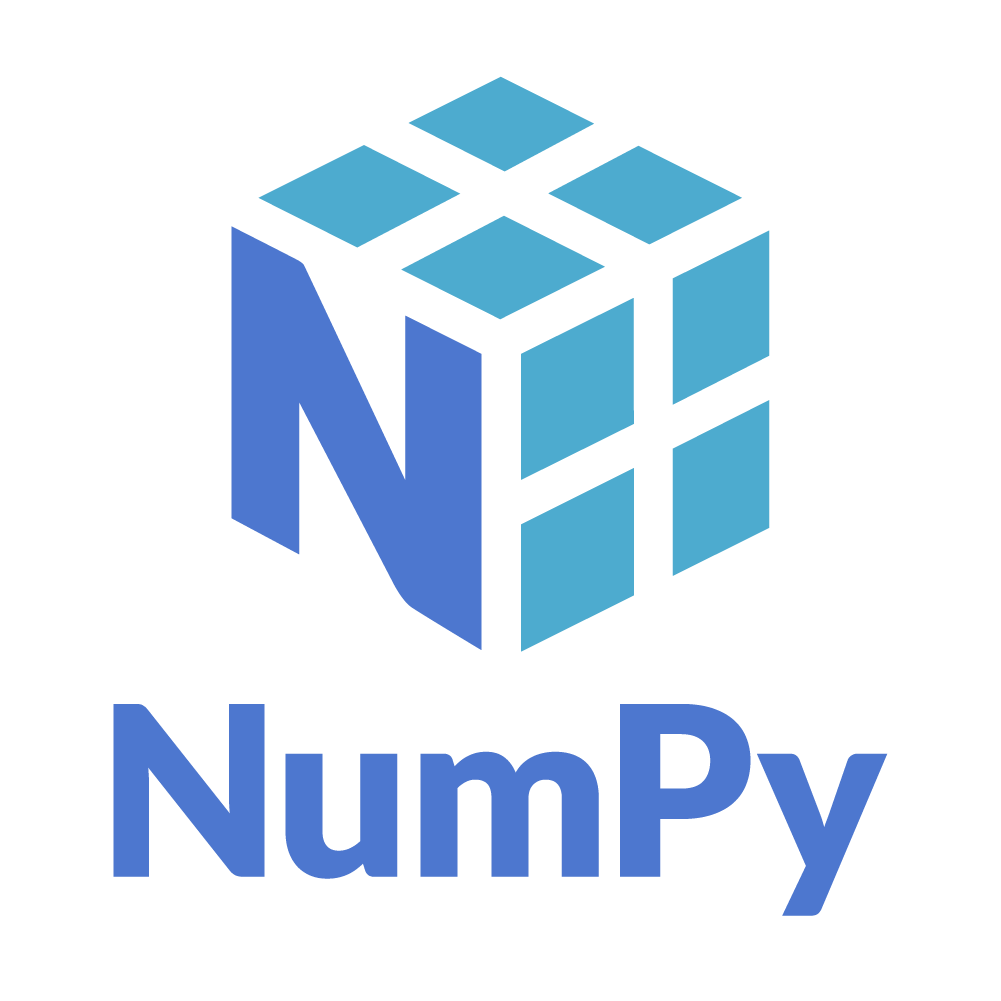
**3.2.3 Dataset Description.** The dataset contains 581888 words from eBook Project Gutenberg's The Adventures of Sherlock Holmes, by Arthur Conan Doyle. The Datasets for text data are easy to find and we can consider Project Gutenberg which is a volunteer effort to digitize and archive cultural works, to “encourage the creation and distribution of eBooks”. From here many stories, documentations, and text data which are necessary for our problem statement can be obtained.

**3.3.4 Import Packages.** Packages are namespaces which contain multiple packages and modules themselves. They are simply directories, but with a twist. Each package in python is a directory which must contain a special file called \_\_init\_\_py. This file can be empty, and it indicates that the directory it contains is a Python package, so it can be imported the same way a module can be imported [28].

The packages used in the proposed system are:

* Numpy
* Pickle
* Tensorflow
* Nltk
* Pydot
* Pydotplus
* Re
* Keras

Syntax for importing a package is Import package\_name as Object\_package\_name

** 3.3.4.1 NumPy.** NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. [29]

**Figure 3.7 NumPy [30]**

**3.3.4.2 Pickle.** Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that character stream contains all the information necessary to reconstruct the object in another python script [31].

**3.3.4.3 Tensorflow.** Tensorflow is a python library used for developing Deep learning application. It was developed by Google primarily. It is an open-source library as it available for free. It also supports machine leaning system. Originally Tensorflow was developed for enormous numerical computations. Tensorflow will receive the data in multi-dimensional array format of higher dimensional are called tensor. It will work on the base of data flow graph which have nodes and edges [32]. The functions used in this project are Embedding, LSTM, Dense, Activation, Tokenizer, to\_categorical, Adam.

**3.3.4.4 Nltk**. **NLTK (Natural Language Toolkit)** Library is a suite that contains libraries and programs for statistical language processing. It is one of the most powerful NLP libraries, which contains packages to make machines understand human language and reply to it with an appropriate response [33]. The functions used in this project are sent\_tokenize, word\_tokenize.

**3.3.4.5 Pydot.** Pydot is a python library, also written in python, that “serves as a graphical interface to Graphviz, open source graph visualization software. GraphViz is written in DOT language, but Pydot provides the ability to parse and dump data, between Python and DOT” [34].

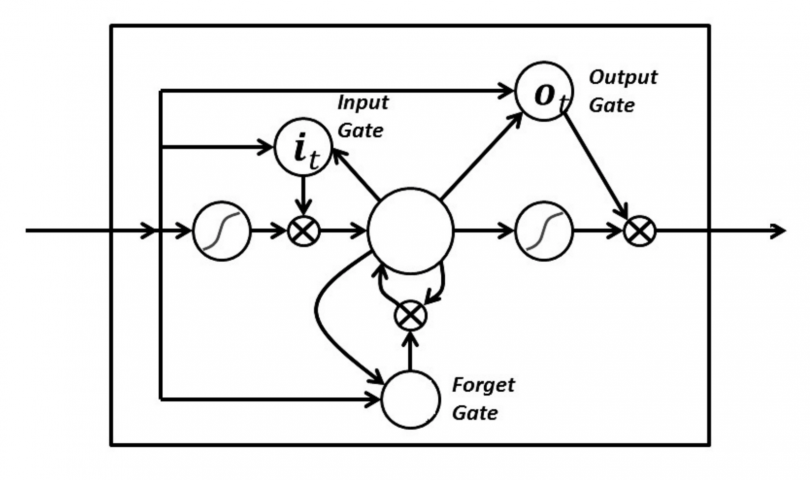
**3.3.4.6 Pydotplus.** Pydotplus is an improved version of the old pydot project that provides a Python Interface to Graphviz’s Dot language [35].

**3.3.4.7 Re.** A regular expression (or RE) specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression (or if a given regular expression matches a particular string, which comes down to the same thing) [36].

**3.3.5 Feature engineering.** This step involves the art and science of transformation raw data into feature that better represent a pattern to the learning algorithms. For example, data can be decomposed into multiple parts to capture more specific relationships, such as analyzing sales performance by the day of the week, not only the month or year. In this situation, segregating the day as a separate categorical value from the date (e.g. “Mon; 06.19.2017”) may provide the algorithm with more relevant information [37].

**3.6 Algorithm used in the study.** The main objective of this project is Prediction the next word for given sentence. The Long Short Term Memory (LSTM) one of the deep learning algorithm is used for next word prediction

**3.6.1 Long Short Term Memory (LSTM).** Long Short-Term Memory (LSTM) network are an extension of RNN extend the memory. LSTM are used as the building blocks for the layers of a RNN. LSTMs assign data “weight” which helps RNNs to either let new information in, forget information or give it importance enough to impact the output. LSTMs enable RNNs to remember inputs over a long period of time. This is because LSTMs contain information in a memory, much like the memory of a computer. The LSTM can read, write and delete information from its memory [38].

This memory can be seen as a gated cell, with gated meaning the cell decides whether or not to store or delete information (i.e., if it opens the gates or not), based on the importance it assigns to the information. The assigning of importance happens through weights, which are also learned by the algorithm. This simply means that it learns over time what information is important and what is not. In an LSTM there are three gates: input, forget and output gate. These gates determine whether or not to let new input in (input gate), delete the information because it isn’t important (forget gate), or let it impact the output at the current timestep (output gate). Below is an illustration of a RNN with its three gates:

**Figure 3.8 Architecture of LSTM [38]**

The gates in an LSTM are analog in the form of sigmoids, meaning they range from zero to one. The fact that they are analog enables them to do backpropagation. The problematic issue of vanishing gradients is solved through LSTM because it keeps the gradients steep enough, which keeps the training relatively short and the accuracy high [38].

* + 1. **Applications.** Some of the most demanding applications are discussed below [39]:
* Language modeling or text generation, that involves the computation of words when a sequence of words is fed as input. Language models can be operated at the character level, n-gram level, sentence level or even paragraph level.
* Image processing that involves performing analysis of a picture and concluding its result into a sentence. For this, it’s required to have a dataset comprising of a good amount of pictures with their corresponding descriptive captions. A model that has already been trained is used to predict features of images present in the dataset. This is photo data. The dataset is then processed in such a way that only the words that are most suggestive are present in it. This is text data. Using these two types of data, we try to fit the model. The work of the model is to generate a descriptive sentence for the picture one word at a time by taking input words that were predicted previously by the model and also the image.
* Speech and Handwriting Recognition
* Music generation which is quite similar to that of text generation where LSTMs predict musical notes instead of text by analyzing a combination of given notes fed as input.
* Language Translation involves mapping a sequence in one language to a sequence in another language. Similar to image processing, a dataset, containing phrases and their translations, is first cleaned and only a part of it is used to train the model. An encoder-decoder LSTM model is used which first converts input sequence to its vector representation (encoding) and then outputs it to its translated version.
  + 1. **Advantages of LSTM.** Some of advantages of LSTM are [40]:
* Non decaying error backpropagation.
* For long time lag problem, LSTM can handle noise, distributed representation, and continuous values. In contrast to finite state automata or hidden Markov models LSTM does not require an a priori choice of a finite number of states. In principle it can deal with unlimited state number.
* No need for parameter tuning. LSTM works well over a broad range of parameters such as learning rate, input gate bias and output gate bias. For instance, to some readers the learning rates used in our experiments may seem large. However, a large learning rate pushes the output gates towards zero, thus automatically countermanding its own negative effects.
* Memory for long time periods.
  + 1. **Disadvantages of LSTM.** Somedisadvantages of LSTM are [41]:
* LSTMs became popular because they could solve the problem of vanishing gradients. But it turns out, they fail to remove it completely. The problem lies in the fact that the data still has to move from cell to cell for its evaluation. Moreover, the cell has become quite complex now with the additional features (such as forget gates) being brought into the picture.
* They require a lot of resources and time to get trained and become ready for real-world applications. In technical terms, they need high memory-bandwidth because of linear layers present in each cell which the system usually fails to provide for. Thus, hardware-wise, LSTMs become quite inefficient.
* With the rise of data mining, developers are looking for a model that can remember past information for a longer time than LSTMs. The source of inspiration for such kind of model is the human habit of dividing a given piece of information into small parts for easy remembrance.
* LSTMs get affected by different random weight initialization and hence behave quite similar to that of a feed-forward neural net. They prefer small weight initialization instead.
* LSTMs are prone to overfitting and it is difficult to apply the dropout algorithm to curb this issue. Dropout is a regularization method where input and recurrent connections to LSTM units are probabilistically excluded from activation and weight updates while training a network [41].

The LSTM algorithm is studied and implemented in the python for word prediction in this project. The step by step results obtained in the implementation stage is discussed in next chapter.

**CHAPTER IV**

**IMPLEMENTATION, RESULT AND DISCUSSION**

**4.1 Implementation and results**

The different implementation stages and the results obtained are discussed in this chapter.

**4.1.1 Importing Libraries.** In python, modules are accessed by using the import statement. It executes the code of the module, keeping scopes of the definitions so that the current file(s) can make use of these.

**Coding**

import nltk

import re

import numpy as np

import tensorflow

import keras

from keras.models import Sequential, load\_model

import tensorflow as tf

from tensorflow.keras.layers import Embedding, LSTM, Dense, Activation

from tensorflow.keras.preprocessing.text import Tokenizer

import pickle

from tensorflow.keras.utils import to\_categorical

from tensorflow.keras.optimizers import Adam

import numpy as np

import os

import pydot

import pydotplus

from pydotplus import graphviz

from keras.utils.vis\_utils import plot\_model

from keras.utils.vis\_utils import model\_to\_dot

keras.utils.vis\_utils.pydot = pydot

**4.1.2 Import dataset.** While running python programs, datasets are used for data analysis. Python has various modules which help in importing the external data in various file formats to a python program. The dataset [27] for this project is imported using the following code.

**Coding**

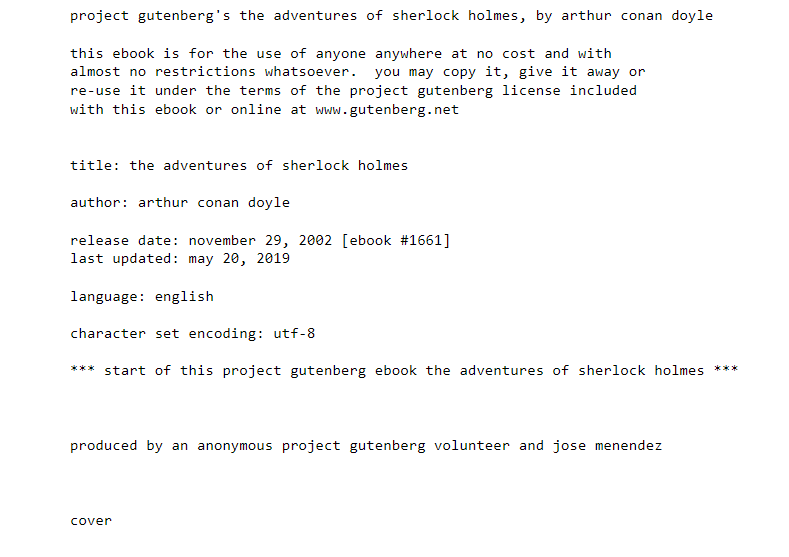
path=open("C:\\Users\\LENOVO\\Downloads\\text.txt","r",encoding='utf-8')

**4.1.3 Data Exploration.** Converting everything to lower case to avoid duplication of words

**Coding**

text =path.read().lower()

print(text)

**Output**

**Figure 4.1 Displaying the text**

The length of the corpus is checked by using the len function on text

**Coding**

print("length of text is:",len(text))

**Output**

The following figure 4.2 shows the length of text is 581888

****

**Figure 4.2 Text length**

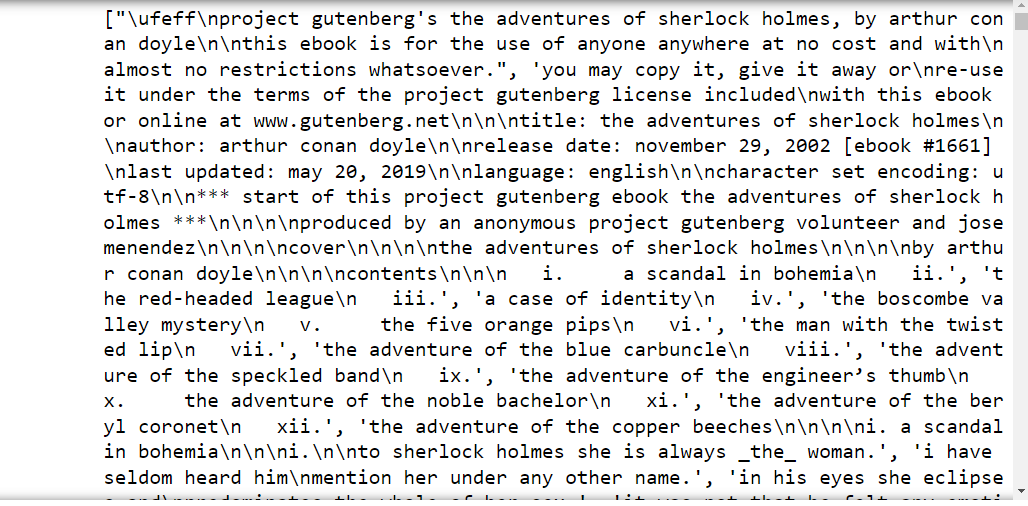
**4.2 Cleaning the text**

NLP Text preprocessing is a method to clean the text in order to make it ready to feed to models. Noise in the text comes in varied forms like emojis, punctuations, and different cases. All these noises are of no use to machines and hence need to clean it [42].In the below code divides a given text into different lines by using the function sent\_tokenize.

**Coding**

sentance=nltk.sent\_tokenize(text)

print(sentance)

**Output**

**Figure 4.3 Sentence tokenization**

Non-number or non-character with space are replaced using the following code.

**Coding**

corpus=[]

for i in range(len(sentance)):

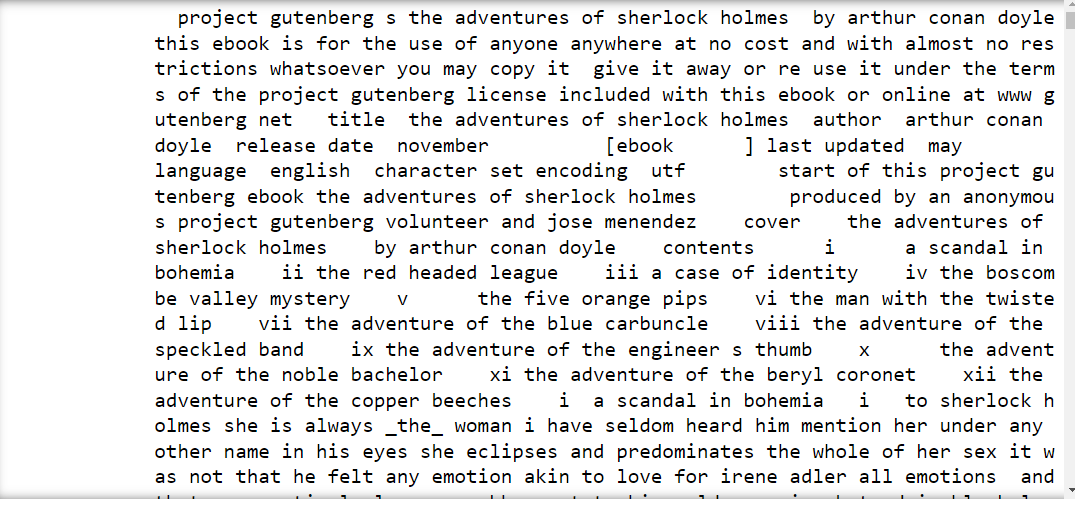
sent=re.sub('[^a-zA-z]', " ",sentance[i])

corpus.append(sent)

corpus="".join(corpus)

print(corpus)

**Output**

****

**Figure 4.4 Cleaned text**

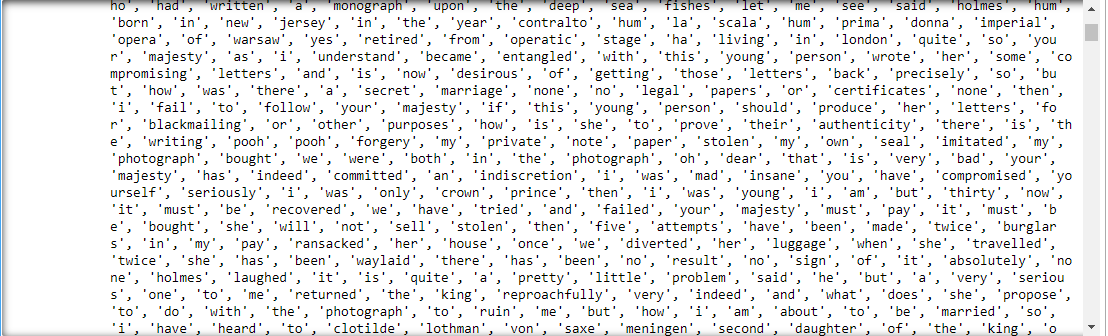
The below code divides a given text into word by using the function word\_tokenize

**Coding**

word=nltk.word\_tokenize(corpus)

print(word)

**Output**

****

**Figure 4.5 word tokenization**

### 4.3 Using tokenizers

The Keras **Tokenizer is** used to vectorize a text corpus, by turning each text into either a sequence of integers (each integer being the index of a token in a dictionary) or into a vector where the coefficient for each token could be binary, based on word count, based on tf-idf [43].

**Coding**

tokenizer = Tokenizer()

tokenizer.fit\_on\_texts([word])

# saving the tokenizer for predict function

pickle.dump(tokenizer, open('token.pkl', 'wb'))

sequence\_data = tokenizer.texts\_to\_sequences([word])[0]

sequence\_data[:15]

**Output**

****

**Figure 4.6 Numeric Representations of Text Data**

The length of the sequence\_data is checked by using the len function

**Coding**

len(sequence\_data)

**Output**

****

**Figure 4.7 Length of sequence data**

### 4.4 Getting unique words:

Calculate the vocab\_size by using the length extracted from tokenizer.word\_index and then add 1 to it. 1is added because 0 is reserved for padding to start the count from 1.

**Coding**

vocab\_size = len(tokenizer.word\_index) + 1

print(vocab\_size)

**Output**

****

**Figure 4.8 Length of vacab size**

### 4.5 Feature Engineering:

Feature engineering will make the words into numerical representation so that it is easy to process them.

**Coding**

sequences = []

for i in range(1, len(sequence\_data)):

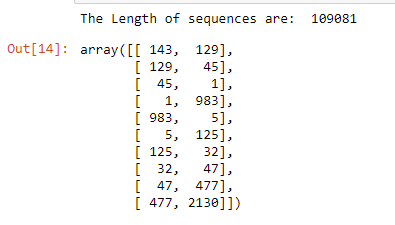
words = sequence\_data[i-1:i+1]

sequences.append(words)

print("The Length of sequences are: ", len(sequences))

sequences = np.array(sequences)

sequences[:10]

**Output**

**Figure 4.9 Matching Input word and next word indices**

### 4.6 Storing features and labels:

The ‘X’ will contain the training data with the input of text data. The ‘y’ will contain the outputs for the training data. So, the ‘y’ contains all the next word predictions for each input ‘X’.

**Coding**

X = []

y = []

for i in sequences:

X.append(i[0:1])

y.append(i[1])

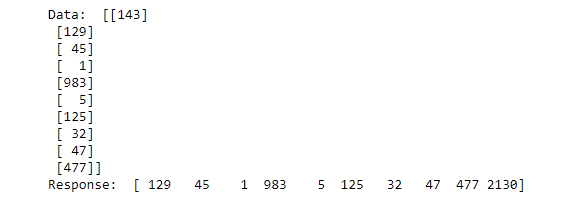
X = np.array(X)

y = np.array(y)

print("Data: ", X[:10])

print("Response: ", y[:10])

**Output**

****

**Figure 4.10 Splited feature and label data**

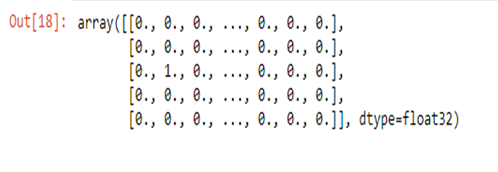
Convert a class vector (integers) to the binary class matrix. This will be useful while using loss function, categorical\_crossentropy.

**Coding**

y = to\_categorical(y, num\_classes=vocab\_size)

y[:5]

**Output**

****

**Figure 4.11 Binary class matrix of labels**

**4.7 Creating the Model:**

The input length of the model is 1, to predict one word as response. The other parameters of the function LSTM are set to default values as shown in the code below. It has 2 layers with 1000 units with relu set as the activation. The output layer is specified with vocab size and softmax activation.

**Coding**

model = Sequential()

model.add(Embedding(vocab\_size, 10, input\_length=3))

model.add(LSTM(1000, return\_sequences=True))

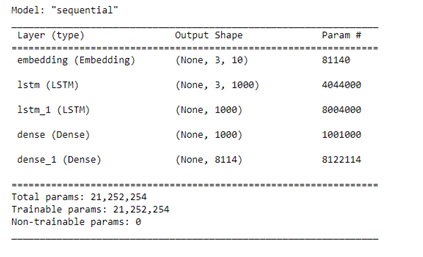
model.add(LSTM(1000))

model.add(Dense(1000, activation="relu"))

model.add(Dense(vocab\_size, activation="softmax"))

model.summary()

**Output**

****

**Figure 4.12 Summary of Model**

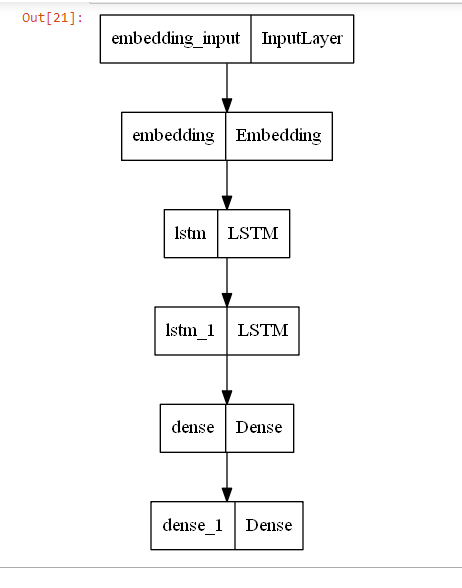
The following code is used to get a model plot.

**Coding**

from tensorflow import keras

from keras.utils.vis\_utils import plot\_model

keras.utils.plot\_model(model, to\_file='plot.png', show\_layer\_names=True)

**Output**

**Figure 4.13 Model Plot**

**4.8 Training and saving the model**

**ModelCheckpoint** callback is used for storing the weights of model after training. It save only the best weights of model by specifying save\_best\_only=True. The model training is monitored by using the loss metric. After training best weights are saved so that re-training of the model is not needed and the saved model can be used when required.  The loss used in the model is categorical\_crossentropy which computes the cross-entropy loss between the labels and predictions. The optimizer is Adam with a learning rate of 0.001 and compile model on the metric loss.

**Coding**

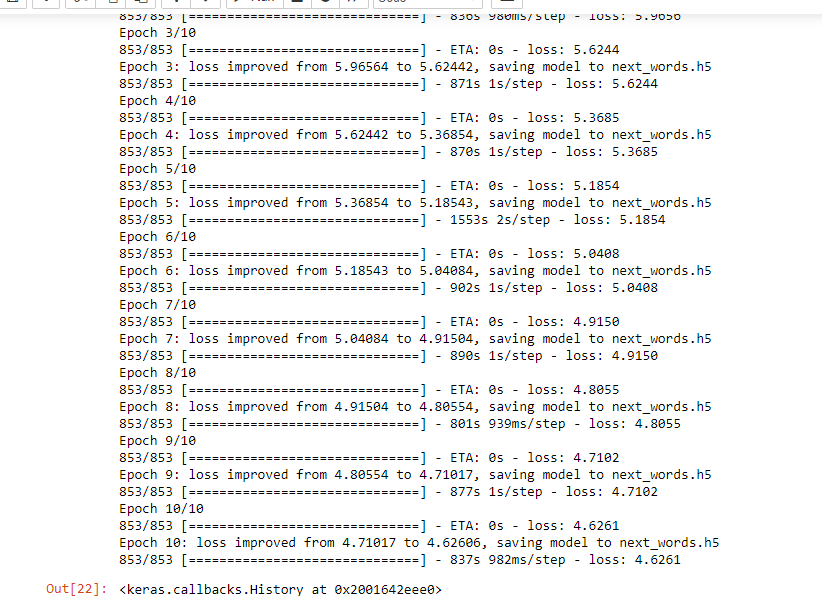
from tensorflow.keras.callbacks import ModelCheckpoint

checkpoint=ModelCheckpoint("next\_words.h5",monitor='loss',verbose=1, save\_best\_only=True)

model.compile(loss="categorical\_crossentropy", optimizer=Adam(learning\_rate=0.001))

model.fit(X, y, epochs=1, batch\_size=128, callbacks=[checkpoint])

**Output**

****

**Figure 4.14 Model Training**

### 4.9 Testing next word

The following code to predict the next few words when we provide a sentence

**Coding**

# Load the model and tokenizer

model = load\_model('next\_words.h5')

tokenizer = pickle.load(open('token.pkl', 'rb'))

def Predict\_Next\_Words(model, tokenizer, text):

sequence = tokenizer.texts\_to\_sequences([text])

sequence = np.array(sequence)

preds = np.argmax(model.predict(sequence))

predicted\_word = ""

for key, value in tokenizer.word\_index.items():

if value == preds:

predicted\_word = key

break

print(predicted\_word)

return predicted\_word

while(True):

text = input("Enter your line: ")

if text == "0":

print("Execution completed.....")

break

else:

try:

text = text.split(" ")

text = text[-1:]

print(text)

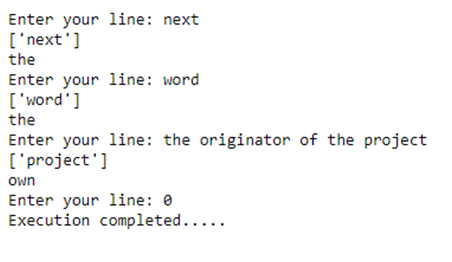
Predict\_Next\_Words(model, tokenizer, text)

except Exception as e:

print("Error occurred: ",e)

continue

**Output**



**Figure 4.15 Prediction**

**CHAPTER V**

**CONCLUSION AND FUTURE WORK**

**5.1 Conclusion**

The main project “NEXT WORD PREDICTION WITH NLP AND DEEP LEARNING” has covered and accomplished the objectives declared. In this project, Long Short Term Memory (LSTM) is implemented using python to predict the Next word for given sentence.

**5.2 Future enhancement**

The future enhancement of this project may create lyrics and to help the end-users to predict the next phrase in songs by training the model on a music lyrics data set. Also, new data/ data preprocessing steps can be explored. Different machine learning algorithms like RNN can also be tired. It can help soon to understand and frame paragraphs and stories on their own.

**REFERENCES**

[1] <https://www.analyticsvidhya.com/blog/2021/08/predict-the-next-word-of-your-text-using-long-short-term-memory-lstm/>

[2]<https://www.itmconferences.org/articles/itmconf/pdf/2021/05/itmconf_icacc2021_03034.pdf>

[3]https://towardsdatascience.com/next-word-prediction-with-nlp-and-deep-learning-48b9fe0a17bf

[4]<https://towardsdatascience.com/next-word-prediction-with-nlp-and-deep-learning-48b9fe0a17bf>

[5]<https://www.techtarget.com/searchenterpriseai/definition/languagemodeling#:~:text=Importance%>

[6]<https://www.researchgate.net/publication/347267084_Natural_Language_Word_Prediction_Model_Based_on_Multi-Window_Convolution_and_Residual_Network>

[7] <https://ieeexplore.ieee.org/document/8955090>

[8] <https://www.mecs-press.org/ijisa/ijisa-v10-n2/v10n2-5.html>

[9]<https://www.researchgate.net/publication/350906250_Pretraining_Federated_Text_Models_for_Next_Word_Prediction>

[10] <https://ieeexplore.ieee.org/document/713907>

[11] <https://ieeexplore.ieee.org/document/6481820>

[12] <https://ieeexplore.ieee.org/document/7477536>

[13] <https://www.sciencedirect.com/science/article/pii/S1877050918320556?via%3Dihub>

[14] <https://www.expert.ai/blog/machine-learning-definition/>

[15] https://www.geeksforgeeks.org/machine-learning/

[16] <https://www.djuices.com/many-type-of-machine-learning/>

[17]<https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a>

[18] <https://www.datarobot.com/blog/semi-supervised-learning/>

[19] <https://www.ibm.com/cloud/learn/deep-learning>

[20] <https://www.geeksforgeeks.org/machine-learning/>

[21] <https://www.techopedia.com/definition/5967/artificial-neural-network-ann>

[22]<https://towardsdatascience.com/introduction-to-neural-networks-advantages-and-applications-96851bd1a207>

[23] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[24] https://builtin.com/data-science/recurrent-neural-networks-and-lstm

[25] <https://www.ibm.com/cloud/learn/deep-learning>

[26]https://www.analyticsvidhya.com/blog/2021/08/predict-the-next-word-of-your-text-using-long-short-term-memory-lstm/

[27] <https://drive.google.com/file/d/1GeUzNVqiixXHnTl8oNiQ2W3CynX_lsu2/view>

[28] https://www.learnpython.org/en/Modules\_and\_Packages

[29] <https://numpy.org/doc/stable/user/whatisnumpy.html>

[30] https://github.com/gilbarbara/logos/issues/428

[31] https://www.geeksforgeeks.org/understanding-python-pickling-example/#:~:text=Python%20pickle%20module%20is%20used,list%2C%20dict%2C%20etc.

[32] <https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-tensorflow>

[33] <https://www.guru99.com/nltk-tutorial.html>

[34] <https://data2int.com/Pydot>

[35] <https://pydotplus.readthedocs.io/>

[36] <https://www.w3resource.com/python/python-regular-expression.php>

[37] <https://www.kdnuggets.com/2018/12/six-steps-master-machine-learning-data-preparation.html>

[38] <https://builtin.com/data-science/recurrent-neural-networks-and-lstm>

[39] <https://www.geeksforgeeks.org/understanding-of-lstm-networks/#:~:text=LSTMs%20provide%20us%20with%20a,)%2C%20which%20is%20an%20advantage>.

[40] <https://slideplayer.com/slide/13047419/>

[41] <https://www.geeksforgeeks.org/understanding-of-lstm-networks/#:~:text=LSTMs%20provide%20us%20with%20a,)%2C%20which%20is%20an%20advanta>

[42]https://towardsdatascience.com/cleaning-preprocessing-text-data-by-building-nlp-pipeline-853148add68a

[43] https://keras.rstudio.com/reference/text\_tokenizer.html