**Predict the next word of your text using Long Short-Term Memory (LSTM)**

**Statement:** Most of the keyboards in smartphones give next word prediction features; google also uses next word prediction based on our browsing history. So, a preloaded data is also stored in the keyboard function of our smartphones to predict the next word correctly. I will train a Deep Learning model for next word prediction using Python. I will use the Tensorflow and Keras library in Python for next word prediction model.

## **Introduction:**



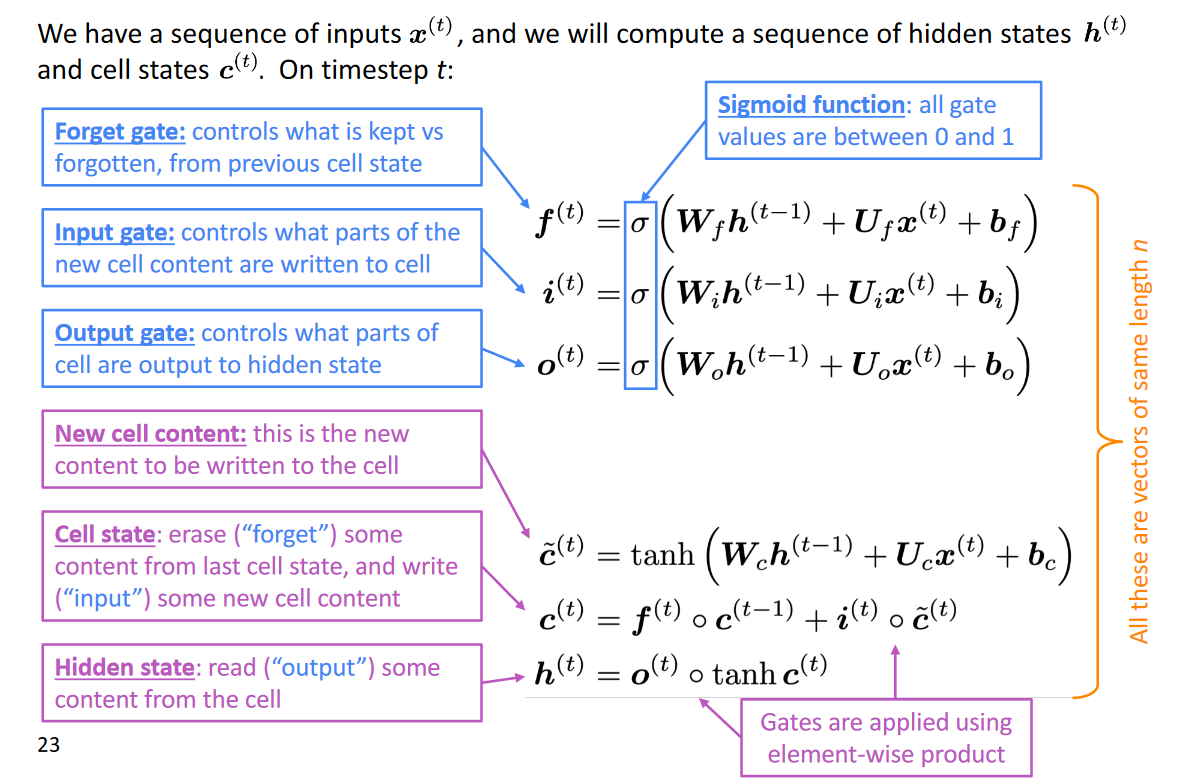
Natural language processing has been an area of research and used widely in different applications. We often love texting each other and find that whenever we try to type a text a suggestion pops up trying to predict the next word we want to write. This process of prediction is one of the applications NLP deals with. We have made huge progress here and we can use Recurrent neural networks for such a process. There have been difficulties in basic RNN.

This Project deals with how we can use a neural model better than a basic RNN and use it to predict the next word. We deal with a model called Long Short-term Memory (LSTM). We can use the TensorFlow library in python for building and training the deep learning model.

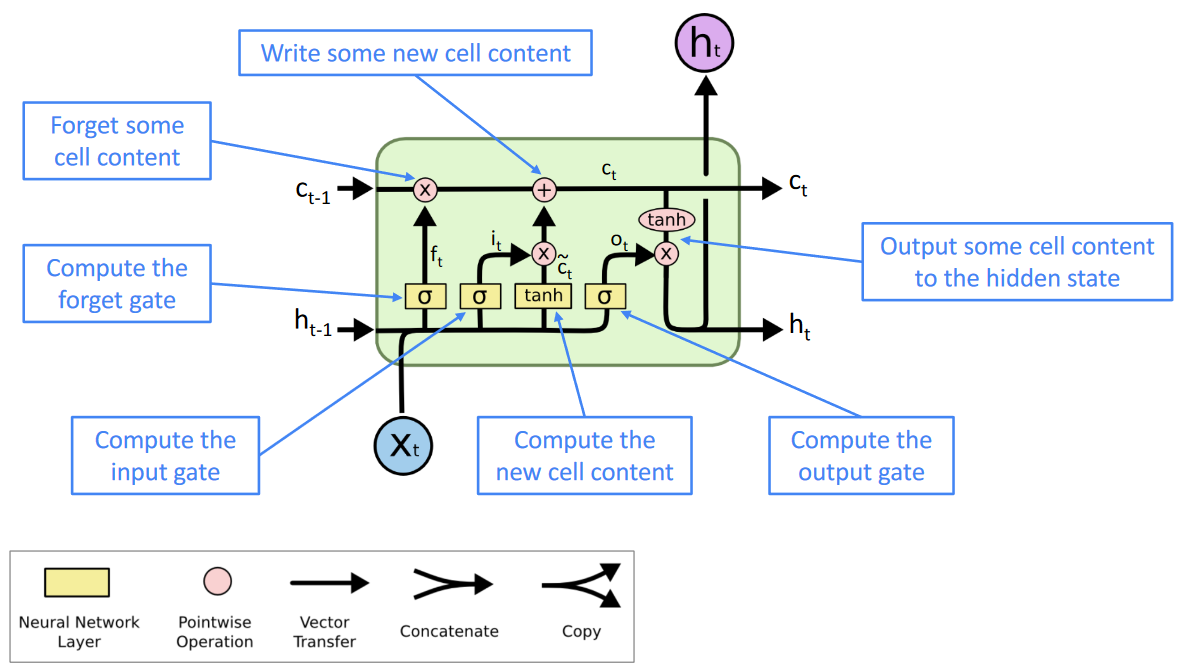
**Proposed Solution**

### **Why use LSTM?**

Vanishing gradient descent is a problem faced by neural networks when we go for backpropagation. It has a huge effect and the weight update process is widely affected and the model became useless. So, we used LSTM which has a hidden state and a memory cell with three gates that are forgotten, read, and input gate.



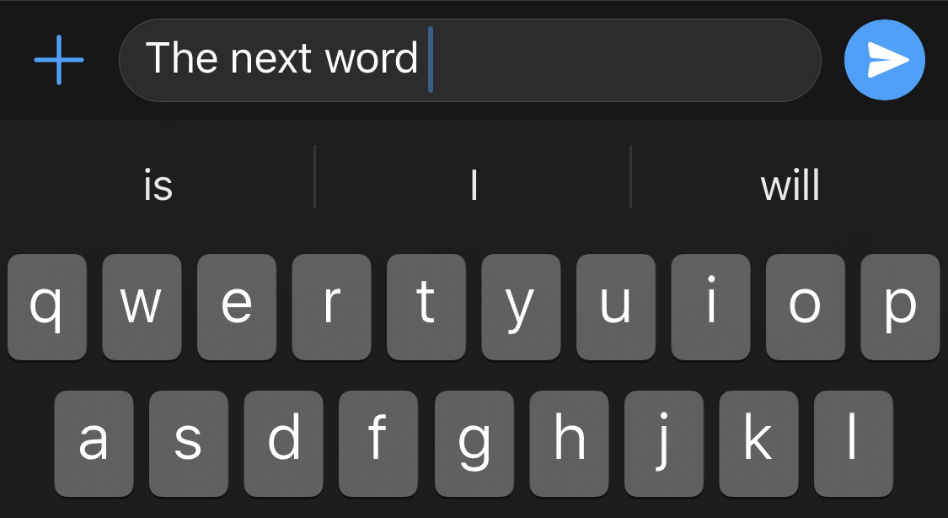
The following figure helps us understand how these gates work. The forget gate is mainly used to get good control of what information needs to be removed which isn’t necessary. Input gate makes sure that newer information is added to the cell and output makes sure what parts of the cell are output to the next hidden state. The sigmoid function used in each gate equation makes sure we can bring down the value to either a 0 or 1.



The exact architecture of an LSTM is shown in this figure. Here, X is the word subscript, t indicates that time instant. As we can see, c and h are input coming from an earlier time or the last step. We have the forget gate that controls the weights so that it can exactly know what information needs to be removed before going to the next gate. We use sigmoid here. The input I have added and some new information is written in the cell at that time instant. Finally, the output gate outputs the information that is given to the next LSTM cell.

## **Prediction of next word:**

Till now we saw how an LSTM works and its architecture. Now comes the application part. Predicting the next word is a neural application that uses Recurrent neural networks. Since basic recurrent neural networks have a lot of flaws we go for LSTM. Here we can make sure of having longer memory of what words are important with help of those three gates we saw earlier.



**Dataset**

[Top 100 | Project Gutenberg](https://www.gutenberg.org/browse/scores/top) refer this link for datasets

**Steps for Implementing Code**

Refer the Jupyter notebook file for the code implementation

**Conclusion:**

Next word prediction is the trending topic in Natural Language Processing (NLP) for last decade. Previously, Support Vector Machines or Markov models were used for next word prediction. With the improvement of technology, NLP model convert to Deep learning algorithms like Recurrent Neural Networks (RNN) and Long Short-Term Memory Networks (LSTM).