**Project Name:** Machine Translation Model

**Github Link:** https://github.com/utkrisht2000/utkrisht2000-Next-Word-Prediction-Model.git

**Why was this project created?**

The technique of translating text from one language into another using a pre-established mapping is known as machine translation. When a user is conversant in a language but not proficient in writing its script, this is helpful. India's national language is Hindi. In India, it is the language that is written and spoken the most. Pronunciation aids for words and names in other languages are provided by machine translation models. Only the letters or characters of the source language should be changed into their equivalents in the destination language, according to the machine translation model. Contrary to translation, which translates the spoken or written meanings of words or texts from one language into another, it does not render meaning.

**What problem is it solving?**

One task where machines unquestionably lag behind human cognitive abilities is language translation. One of the traditional approaches to addressing the machine translation issue is through machine translation. This technique works effectively on pairs of structured languages with similar syntax but requires enormous data sets. Neural machine translation has become a different approach to solving the same problem in recent years.

**Entire explanation of project**

* **PROPOSED APPROACH**

The IIT Bombay English-Hindi Corpus is one of the most comprehensive corpora accessible for carrying out English-Hindi translation tasks, hence I utilised it as the dataset for the course. Pre-processing entails stripping out punctuation, numbers, excess spaces, and changing capitalization to lowercase. The argument oov token, which effectively means "Out-of-Vocabulary," should be taken into consideration. This is particularly useful if we wish to limit our vocabulary to, say, the top 5000 terms. In that case, the parameter value would be used to replace any unpopular words (rank of word based on instance count > 5000) and they would be deemed as not being part of the vocabulary. Each word in the input sequence is taken in by the encoder block. Since we are employing an LSTM layer, the internal state vectors [h, c] will be modified with each word that is consumed. We only care about the final values of these vectors that will be sent to the decoder after the final word. The following phase consumes the encoder's final [h,c] vector as well as each token of the target sentence starting with "START." Each token's output will be compared to the predicted output, and error gradients will be generated to update the model. When the target inputs have been used up, we anticipate our model to generate an “END” token signifying the end of this data sample. The internal states of the encoder are used to configure the decoder LSTM, and its output is fed into a dense layer with a softmax activation and a number of units corresponding to the target vocab size (hindi vocab size). A probability distribution should be created for the entire target vocabulary. Choosing the token with the highest value indicates that it has the highest probability, which is exactly what we do in this case. As the loss function to be optimized, using SparseCategoricalCrossEntropy.

Algorithm for creating next word prediction model :

**Step 1:** Dataset is imported

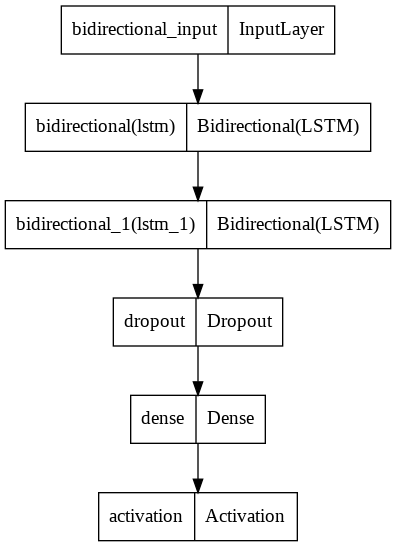
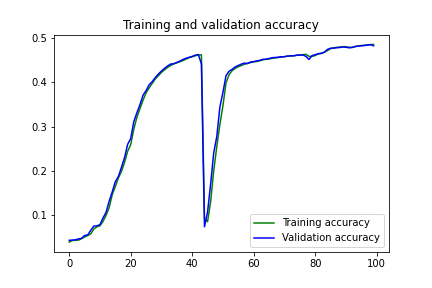
**Step 2:** The data is preprocessed, and the dataset is split into training and testing.

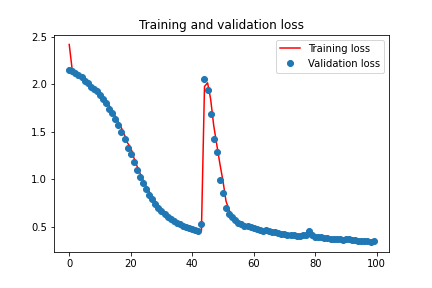
**Step 3:** Text Analysis

**Step 5:** Prepare Decoder Inputs and Outputs

**Step 6:** Create LSTM model

**Step 7:** Compile and Fit

* **DATA FLOW DIAGRAM**
* **RESULT**



* **CONCLUSION**

The encoder is in charge of reading a word from the source language and encoding it to an internal representation in an encoder decoder model. Using the encoded representation of the source language, the decoder is a language model that produces the word translation in the target language. Each step in the output is generated in the context of the complete encoded input.