Word Level English to Hindi Machine Translation using Encoder-Decoder LSTMs

Ajith Kumar, Karthik
ajithkumar@uni-potsdam.de
Master's Program in *Cognitive Systems*
University of Potsdam

Abstract India hosting over 1.3 billion out of 7.5 billion of the world's population, is diverse in culture, tradition and language. India comprises of eighteen constitutional languages where Hindi native speakers are around 41% of the country's population. Majority of this 41% do not know a secondary language like English or understand any other language other than Hindi. For a country like India, this is a trivial situation to deal with in many aspects like Terms and Conditions of various Multi-National Companies where all the documents are usually in English, medical support where the people will not be restricted to very few opinions on the internet or journals. This project is implemented with a strong belief that this project with some modifications could solve a real time issue of serving over five hundred million native speakers of the Hindi Language.

1 Introduction

English to Hindi translation have been performed in many various statistical algorithms in the earlier years. This paper discusses the implementation of a Neural Machine Translation task using word level seq2seq model. The data used for this implementation is a parallel corpus of over 1.4 million sentences in both English and Hindi, provided by Indian Institute of Technology, Bombay. This implementation uses high level deep learning framework Keras on top of Tensorflow to develop a LSTM encoder decoder model. Evaluation of results can be done using BLEU scores. The implementation consists of the BLEU score calculation for the test data. Below diagram will give us a good understanding of the flow of the project(created using draw.io).

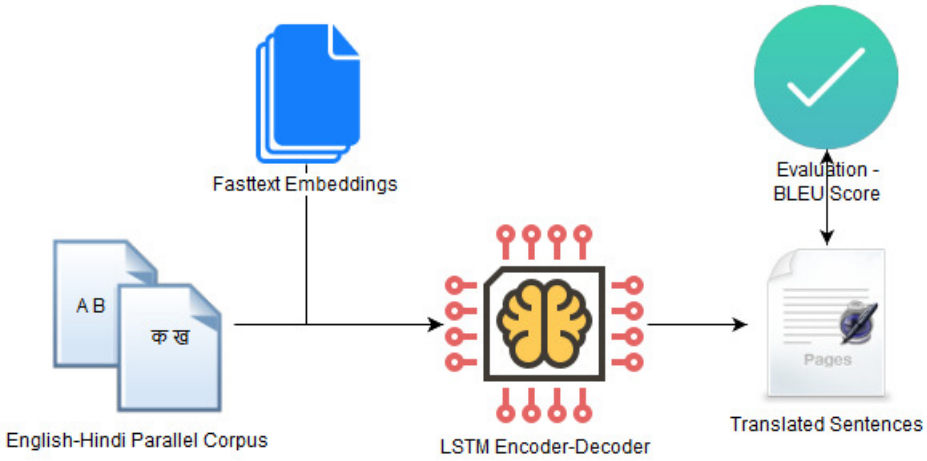


Figure 1: Flow of the project.

The purpose of this paper is to assist those people who are not aware of the mainstream international language, English. With some more future work, one can achieve the best out of this implementation in serving the purpose.

Below are few highlights of the project, methods used, expected results from the NMT model.

– Word level sentence processing: For a language like Hindi, word level processing seems to be a more reliable as Hindi is a language where vowels are added to a consonant which forms a new letter while writing. Consider the below example.

क (k) + आ (aa) = का (kaa)
म (m) + इ (e) = मि (me)

Hindi Language is known to have 13 vowel sounds and 33 consonant sounds. This would be 429 representations of different sounds provided by the language.

– Furthermore, this project implementation uses pre-trained embedding provided by fastText, which are most occurring words from English and Hindi of the vector size of 300 each. Below is examples for each English and Hindi embedding.

the : array([-5.170e-02, 7.400e-02, -1.310e-02, 4.470e-02, -3.430e-02, 2.120e-02, 6.900e-03, -1.630e-02, -1.810e-02, -2.000e-03, -1.021e-01, 5.900e-03, 2.570e-02, -2.600e-03, -5.860e-02, -3.780e-02, 1.630e-02, 1.460e-02, -8.800e-03,....],dtype=float32)

मानव : array([-1.590e-02, 8.930e-02, 9.700e-02, -3.870e-02, 2.930e-02, 3.280e-02, 5.890e-02, 5.280e-02, 1.000e-04, -2.360e-02, 1.580e-02, -8.040e-02, -1.500e-03, -3.070e-02, 1.550e-02, 4.140e-02, -1.990e-02, -3.010e-02, -6.890e-02,....],dtype=float32)

2 Related Work

Over the years, machine translation tasks from English to Hindi has always been carried out by using various different techniques. Interlingual machine translation is one of the widely know machine translation where the source language is converted to an abstract language called interlingua which is later converted to the target language. This follows a rule-based approach to translating from one language to another. Interlingua is considered better than the Direct Translation which is the direct dictionary-based translation. Transfer-based is the

most well known and most used them in recent years which comprises of three main steps, first, analyses the source language to determine the grammatical structure, second, transferring the generated structure to the target language and then followed by generation of text in the target language. Most recent papers are based on Statistical Machine Translation techniques. SMT uses human translated data, a parallel corpus, to translate from source to target. Unlike the old methods, SMT also accommodates phrase-based translation. A team from IIT Patna has worked on English to Hindi translation using Hierarchical-phrase based SMT using the same parallel corpus that is used in this project implementation. Hierarchical phrase-based SMT was implemented at the Workshop on Asian Translation.

This implementation uses modern methods like deep neural networks for machine translation rather when compared to Statistical Machine translation. Researchers from the University of Zürich claim that Neural Machine Translation outperforms Statistical Machine Translation in three main reasons stating why NMT is a breakthrough(give reference)[1].

- Captures the similarity between words.
- NMT model assesses fluency of the output.
- NMT learns complex relationships between languages.

Said the advantages, there are few disadvantages of NMT over SMT as discussed by the researchers at the University of Zurich. NMT cannot translate a corpus but only sentences, as context learning itself is a problem being solved by Deep Learning. Another drawback of NMT is, it requires a lot of data. Though there is a lot of data available on the internet, gathering a parallel corpus and checking its correctness is a manual task, unlike other language models.
Approach taken in this paper is very different from already existing papers as the primary reason being the usage of Deep learning technologies which has not been explored much for English to Hindi translation. Furthermore, using pre-trained embedding would add weights to the learning model to increase the accuracy, in turn reducing loss, of the model.

3 Project

3.1 Data Gathering and Processing

For a machine translation task, we require a parallel corpus consisting of the source to target language sentences. Data used for this project is a parallel corpus[3] provided by IIT-Bombay, the highest collection of English to Hindi translated sentences summing up to 1,492,827. These parallel sentences are collected from various sources, to name a few, Ted talks, GNOME Opus, Book Translations (Gyaan Nidhi Corpus) etc.

Considering sentences with length less than or equal to 65 in both English and Hindi sentences so to reduce unnecessary noise in the data. After the elimination of these sentences, we get 1,470,000 sentences, which is around 96-97% of the whole data. Followed by adding the prefix of 'START_' and suffix of '_END' for the target language, i.e., all the Hindi sentences.

Along with the parallel corpus, fastText embedding for both English and Hindi is used which is generated using CBOW (Continuous Bag of Words) model with position-weights. Each vector in this embedding is of dimension 300, and double type, generated with n-gram of length 5. The total length of English embedding is two million and Hindi over 1.87 million.

From the parallel corpus, English and Hindi vocabulary was found to be around 240 thousand and Hindi to be around 320 thousand are extracted. Cut down the vocabulary from English words and Hindi words to 80 thousand each, which are most frequently occurring instances. This is done so as to avoid overloading the GPU memory which generating data for the input for the model. Rather than using all the embedding vectors, only the ones matching with the vocabulary is mapped and used as an input for the embedding layer.

3.2 Model Configuration and Training

For a neural machine translation task, most apt architecture is the Recurrent Neural Networks as it is good with time series data. An NMT task is a time series problem where there is a timely occurrence of words in order to learn a sequence.

In the recent past variants such as Gated Recurrent Units (GRU) and Long-Short Term Memory (LSTM) has been a breakthrough for various tasks like language modelling, image captioning, speech recognition and machine translation. GRUs and LSTMs were introduced to overcome the long term sequential dependencies. Between GRU and LSTM, LSTM seems to work better with longer sequences and so this implementation used LSTMs.

To understand how LSTMs work, below are the equations for every gate available in each LSTM cell. The important components are input sequences (x_t), forget gate's vector (f_t), input gate's vector (i_t) and output gate's vector (o_t), weights in our case, pre-trained embedding W, hidden and memory states h_t and c_t respectively, where initial values of c_0 and h_0 are zeros[5][4].

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \tag{1}$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{2}$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{3}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \tag{4}$$

$$h_t = o_t \circ \sigma_h(c_t) \tag{5}$$

The implementation as said earlier is a seq2seq, Encoder-Decoder LSTMs model. To configure the Encoder part of the model, starting from the Input layer which is kept empty, followed by the Embedding layer where the pre-trained embedding that was prepared for all the English vocabulary is used. Followed by an LSTM Layer with a latent space of 512, considering the vocabulary of eighty thousand tokens and over 1.1 million sentences and maximum length of sentences being 65.

Configuring decoder is very similar to that of the encoder as we intend to provide pre-trained embedding for decoder part of the model as well. An empty Input layer followed by the Embedding layer with Hindi embedding, followed by LSTM layer with inputs taken from the output hidden and memory states from the encoder. This is fed into the "soft-max" dense layer, below formula represents the functionality of a soft-max function. Decoder expands the latent space to the larger dimension of the target data, so the decoder output and hidden

and memory states consist of a large matrix.

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Post configuration of the model, time to train it with the data that's been processed. Index representation of both the English and Hindi sentences are obtained which will be the input for training our model. Below shown is the model how the English sentence is sent to the LSTM layer and how it is translated to Hindi.

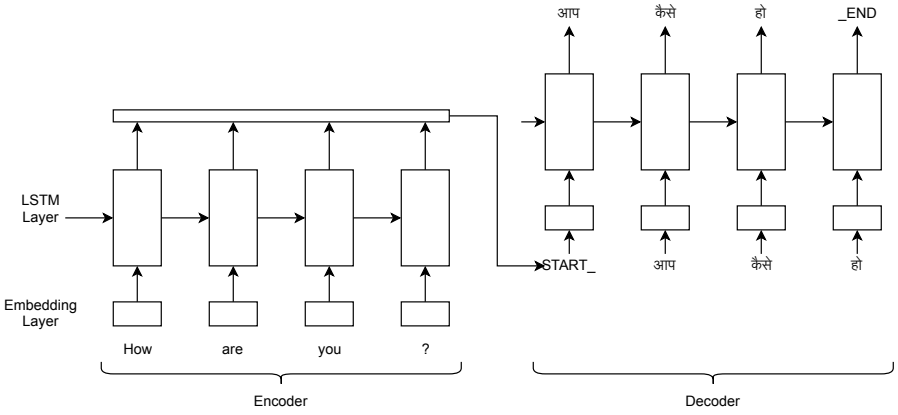
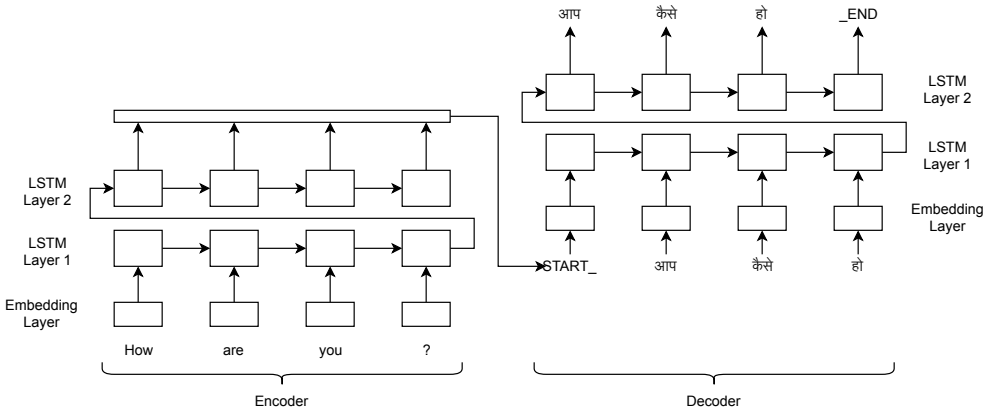


Figure 2: Caption

Just training the model is not sufficient for a seq2seq problem. To decode the sequence for a test sentence, the inference has to be made using the encoder and decoders' output states. At first, the implementation contained only one LSTM layer in both the encoder and the decoder[2]. But for better learning, two LSTMs where the output of the first LSTM is fed to the second LSTM as a sequence is incorporated. Below shown is the multiple LSTM model.

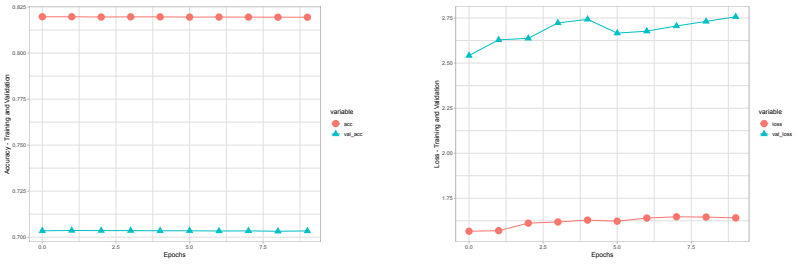


4 Evaluation

The expected result of this project is a Hindi sentence, given an English sentence as an input which infers the output from the output states of the LSTMs. Consider the below example of sample input and output.

Input: *He is a student.*
Output: वह एक छात्र है।

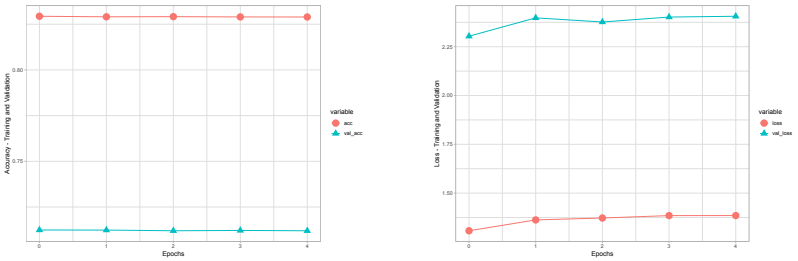
Accuracy obtained by the model is shown in the below graph. There is not much of a variation as the model was run for five-ten epochs.



(a) Accuracy (b) Loss

Figure 3: This is observed for a batch size of 64 and a 10 epochs

There is a slight fluctuation in loss but not very significant. Overall this is a basic seq2seq architecture which can be further improved by adding attention.



(a) Accuracy (b) Loss

Figure 4: This is observed for a batch size of 32 and a 5 epochs

There is vast scope for advancement in this project running the same model on a higher end infrastructure with higher memory which can accommodate more vocabulary and longer sentences. Tuning the model parameters while training for better learning of the model. For example, running for higher epochs as seq2seq models take a lot of time to train. Considering a higher batch size which will provide more data and time for the model to learn.

References

1. 3 reasons why neural machine translation is a breakthrough. https://slator.com/technology/3-reasons-why-neural-machine-translation-is-a-breakthrough/. Accessed: 2019-02-28.
2. Neural machine translation using word level seq2seq model. https://medium.com/@dev.elect.iitd/neural-machine-translation-using-word-level-seq2seq-model-47538cba8cd7. Accessed: 2019-02-25.
3. Pushpak Bhattacharyya Anoop Kunchukuttan, Pratik Mehta. The iit bombay english-hindi parallel corpus, 2017.
4. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory, 1997.
5. Wikipedia contributors. Long short-term memory — Wikipedia, the free encyclopedia, 2019. [Online; accessed 3-March-2019].