CS779 Competition: Hindi to English Machine Translation System

Arnav Singla
roll no. 210188
{arnavs21}@iitk.ac.in
Indian Institute of Technology Kanpur (IIT Kanpur)

16 April 2023

Abstract

The problem was basically a machine translation problem where given a hindi sentence we had to convert it into english. My approach was mostly based on Seq2Seq Models, I tried used GRU's(Gated Reccurent Units) and Transformers as my encoders and decoders, both of which were not able to perform very well in the task. BLUE score was used as the primary evaluation metric. test_phase_blue_score(0.025)

1 Competition Result

Codalab Username: A_210188
Final leaderboard rank on the test set: 23
BLEU Score wrt to the best rank: 0.101

2 Problem Description

We are given a challenge where we need to convert hindi sentences into a english sentence. We had to make a neural translation machine which can do the above task. We were given a huge training corpus to do the same.

3 Data Analysis

The train dataset consisted of 140,000 hindi-english sentence pairs and the test set contained 20,000 hindi sentences that had to be converted to english sentences. The training dataset had both english and hindi sentences, so i basically build two seperate vocabularies one for each. The size of hindi vocabulary was around 27939 and that of the english vocabulary was around 24261. The most common words in training set in english vocabulary were 'the', 'and', 'i', and 'of' mostly being stop words.

4 Model Description

I tried Two approaches one involved GRU's and one involved Transformers. Both were based on SEQ2SEQ models. I used cross-entropy loss as the loss function

4.1 GRU's

This model involved a decoder and encoder blocks both being GRU's(Gated recurrent units) each consisted of 6 layers of GRU's stacked on top of another through which the input was passed one token at a time for the batch. The forward pass involved converting the sentences to tensor embeddings using a input embedding layer which will be trained as well and then the encoder encodes the sentences using 1 token at a time, the hidden state of the encoder is then used as a the starting hidden state of the deocder which then produces outputs given the target sequence one token at a time.

4.2 Transformers

This model also used a encoder and a decoder blocks but now both were transformers having 6 layers of encoders and deocders respectively each having 8 multi-head attention blocks. Forward pass is almost the same as the GRU model discussed above. First the input_sentence is converted into tensors using input embedding layer(dim=200) then it is encoded using TransformerEncoder, whose outputs are fed into the TransformerDecoder along the target sequence which is converted to tensor using output embedding layer(dim=200, also present in forward pass, trained along the way) to give the complete target sequence

5 Experiments

5.1 Data Pre-processing

for the hindi sentences the indic-nlp library was used for pre-processing, given any hindi sentence first i converted it to it's lower case then normalized it using the indicnlp.normalize (removing nuktas, etc) then i tokenized it using indicnlp tokenizer itself. Almost the same procedure was followed for english output sentneces as well, just the spacy library was used for this one and ¡SOS¿ and ¡EOS¿ tokens were added at the begining and the end of the sentences to indicate the start and the end of the sentence respectively so that the model can understand what kind of words are used to start a sentence and what kind of words are used for the ending part. Having converted both the input and output sentence to their respective tokens i padded or truncated the sentences to a fixed maximum length(64 in this instance).

5.2 Training Procedure

Both the models were trained on kaggle for about 10-15 iteration each taking time of about 20-30 minutes per epoch with a batch size of 32. I tried using various learning rates ranging from 1e-2 to 1e-7, but the optimal range was around 1e-4. I used Adam optimizer as my primary optimizer to minimize the loss function.

5.3 Hyperparameters

The hyperparameters mostly included the embedding dimension, number of layers to use in the encoder and decoder, learning rates. I ended up using an embedding dimension of 100 for the GRU's and 200 for the transformers looking at their performances. As for the number of layers i was using anywhere from 4-6 layers in both the encoders and the decoders in both the approaches and i used 8 multi head attention blocks in each layer of the transformer. I found the optimal learning rate to be around 3e-4

6 Results

Models	BLUE Score
Transformers	0.025
GRU's	0.021

Table 1: Results on the testing phase

7 Error Analysis

Both the models were repeatedly giving the same word a number of times like 'faithful', 'the' etc and was not able to predict the sentence upto a correct length. GRU's were mostly giving predictions till the maximum length

8 Conclusion

I found the transformers to perform better, but still the training had some issues leading to very poor BLUE score. Keeping in mind future work, we can work on the training process to truly uncover the potential of both the GRU's and the Transformers