Build Your Own Transformer - Software Project Report

Nesara Belakera Lingarajaiah, 7002967

Abstract

The most important objective of language models is the programmed interpretation of content from one language to another.This machine translation is challenging given the inherent ambiguity and flexibility of human language. Neural machine translation models fit a single model rather than a pipeline of fine-tuned models and currently achieve state-of-the-art results. Here, we recreate one such Neural machine translation model called Transformer which is mainly based on the attention and self attentions. This manuscript briefs the reader about how a simple Transformer works and performance of the same depending on the change in hyperparameters such as number of layers, batch size, input and output dimensions, step size of optimizer etc. For simplicity PHP corpus is used and as a result the experimental results are binded to the corpus test sentences alone. Due to this reason the implemented model is not expected to generalize well although it could generalize well if tried on a different Dataset for example like WMT.

1 Introduction

In sequence-to-sequence problems such as the neural machine translation, the initial methods were based on the use of RNNs(recurrent neural network) in an encoder-decoder architecture. These architectures have a great limitation when working with long sequences. Their ability to retain information from the first elements was lost when new elements were incorporated into the sequence.In the encoder, the hidden state in every step is associated with a certain word in the input sentence, usually one of the most recent. Therefore, if the decoder only accesses the last hidden state of the decoder, it will lose relevant information about the first elements of the sequence. RNNs not only have these above-mentioned issues but also have many other limitations such as vanishing or exploding gradient problems, slow and complex training procedures, difficulty to process longer sequences. The particular long term memory issue is that It cannot process very long sequences if using tanh or relu as an activation function. This gave rise to LSTMs(Long Short-Term Memory Network). LSTM networks are a type of RNN that uses special units in addition to standard units. LSTM units include a 'memory cell' that can maintain information in memory for long periods of time. This memory cell lets them learn longer-term dependencies.STMs deal with vanishing and exploding gradient problems by introducing new gates, such as input and forget gates, which allow for a better control over the gradient flow and enable better preservation of "long-range dependencies". Although LSTMs could fulfill few limitations led by RNNs, they too had few limitations such as mainly sequential processing of sentences which must be processed word by word. This led to longer training time and required more memory to train.

In 2017, Transformers (Attention is all you need) were introduced in the context of machine translation with the purpose to avoid recursion in order to allow parallel computation (to reduce training time) and also to reduce drops in performance due to long dependencies.The main characteristics are:

1. **Non sequential:** sentences are processed as a whole rather than word by word.This is the main reason why they do not suffer from long dependency issues. They do not rely on past hidden states to capture dependencies with previous words. They instead process a sentence as a whole. That is why there is no risk to lose (or "forget") past information

2. **Self Attention:** used to compute similarity scores between words in a sentence.

3. **Positional embeddings:** another unit intro-

duced to replace recurrence. The idea behind this is to have information related to a specific position of a token in a sentence.

Hence it is a general practice to use the Transformers network in the current industry of Machine Translation. In this project, Transformers are used to translate a PHP corpus DE-EN using Greedy decoding scheme. Further explanations on the model, data preprocessing and the experimental results can be found in the following sections.

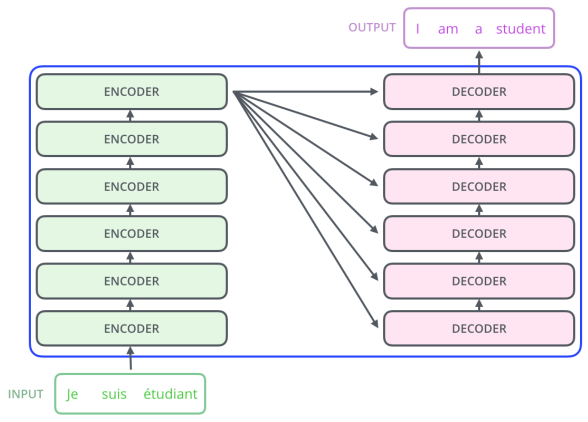

Figure 1: High Level Transformer

2 Model Architecture

Transformer is a type of sequence-to-sequence models which uses a typical encoder-decoder architecture where the encoder comprises of encoding layers that handle the input iteratively one layer after another, whereas the decoder comprises of interpreting layers that do the same thing to the output from encoder. Fig 1 is an illustration of a transformer model. The other major elements are discussed below.

1. **Self-Attention:** It is an attention mechanism relating different positions of a single sequence(sentence) in order to compute a representation of the same sequence (sentence).Every input vector is used in three ways in this mechanism: the Query, the Key and the Value. Three weight matrices (K, Q, V learnable weight layers) of dimensions k x k are computed having three linear transformations for each input vector. These three matrix results come from the same input, on which we apply the attention mechanism of the input vector with itself, a "self-attention". So now the input consists of queries and keys of dimension d_k, and values of dimension d_v using which we compute the dot product of the query with all keys, divide each by the square root of d_k(scaled factor to stabilize gradients), and apply a softmax function to obtain the weights on the values. Finally have the attention scores which represents how much focus to place on other places or words of the input sequence w.r.t a word at a certain position.

2. **Multi-head Attention:** This is basically to attend to different segments of the words. This gives the self attention more power of discrimination between the words of sentences by combining many self attention heads followed by dividing the words vectors into a fixed number (h, number of heads) of chunks, and then self-attention is applied on the corresponding chunks, using Q, K and V submatrices.This produce h different output score matrices. These matrices are concatenated and multiplied by an additional weights matrix. This final matrix captures information from all the attention heads which then serves as an input(one matrix) to the Feed-Forward layer.

3. **Positional Encoding:** As the network and the self-attention mechanism is permutation invariant, the order of the words in the sentence is difficult to solve in transformers. It is invariant to shuffling of words. Hence, positional representation of the words is important and should be added to the word embeddings in both encoder and decoder.So, sinusoidal function is applied to map the position in the sentence to a real valued vector and the network learns how to use this information.

All these main components make the encoder and decoder blocks of a Transformer as in the Figure 2. In the encoder, we have 2 main sublayers, i.e. one Multi-Head attention layer and one feed forward layer. The input of the encoder is the embedding of the sequence itself along with the positional encoding. The decoder is very similar to the encoder but we have one additional sublayer, the Masked Multi-Head Attention layer. The decoder will "hide" future inputs to ensure that a prediction made at a particular time only depends on what is known prior to it. The output of the encoder is given as an input to the decoder. The implementation has 6 such identical parts stacked on one

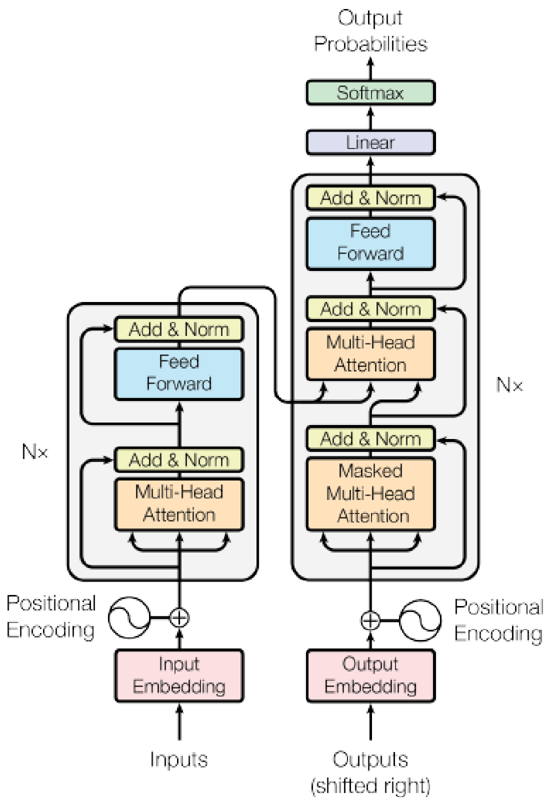

Figure 2: Transformers Architecture

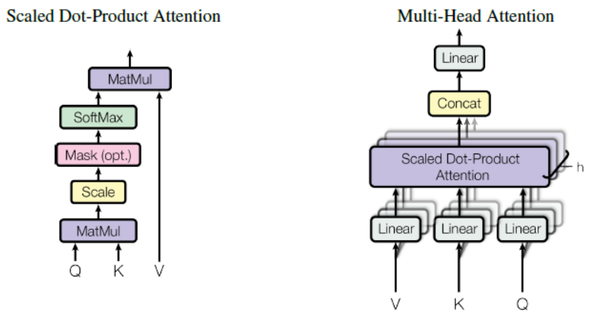

Figure 3: Scaled Dot Product Attention

another.

3 Experimental Setup

The model uses PHP German to English parallel corpus for the experiment. The corpus is rather noisy and may include parts from the English original in some of the translations and as well as in the source sentences (German). The corpus is tokenized and each language pair has been sentence aligned. It is trained for 15 epochs having a batch size of 30 due to resource constraints. The PHP corpus has an initial size 39707 and after removal of duplicates, it is 12281 number of sentences. The issue in removing the duplicates was that the number of duplicates in German language were different from the English. Hence, firstly, mapping of corresponding sentences is done using Lists and Dictionaries and finally the duplicates corresponding to the source (simultaneously mapped with target) are removed. We use the unigram SentencePiece model for subword segmentation and to predetermine the number of unique tokens and the vocab size. Two sentencepiece models are implemented creating two vocab files where each model and vocab files are for source language and target language. The size of the source vocab is 13523 and that of target(English) is 9811. The corpus is randomly split into train, test and validation sets in the proportion 7:2:1. The maximum sentence length is taken to be 200 and index of pad token, start token and that of end token (pad, sos, eos) is set to be 0, 1 and 2. The index of unknown tokens is set to be 3.Using these, the input sentence is converted into a vector of integers of length 200 and fed into the encoder. There are 6 numbers of layers in the encoder and the decoder. The size of hidden states in the model is set to 512 after checking the other possible alternatives such as 128 and 256 as 512 works better. The number of heads for Multi-head attention is set as 8. We use the negative log likelihood loss as a loss function. The Adam optimizer worked the best,hence used Adam with a learning rate of 1e-4. Experiments are performed for dropout rates ranging from 0.1 to 0.3. IOU(intersection over union) is used as a metric to measure the performance of the model. All the hyperparameters can be recorded in the table 1.

4 Results

The transformer model's training and validation curves can be seen in the Figure 4. As we see,

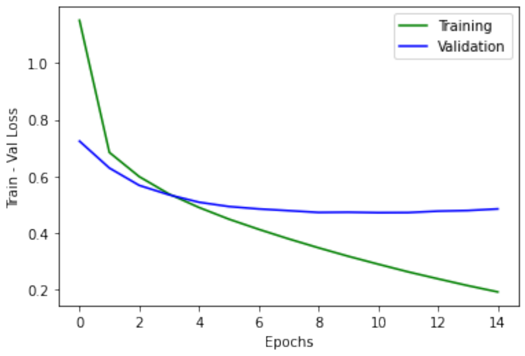

Figure 4: Epochs Vs Loss Curves

the training loss decreases along a period of time but when tested on a validation set, the loss decreases to a certain extent, although not as much as the training curve. Validation curve remains constant and does not decrease any further. The best model's checkpoint is saved for future use. Although there is a considerable amount of good loss found in the validation set, the generalization gap still persists. In future, we could close this gap by tuning the hyper parameters furthermore and increasing the number of layers.The best Training loss is reported to be 0.2250 and that of Validation is 0.5008. After using the Greedy search method, the accuracy i.e. IoU score of the model is found to be 12.03100. Finally, we could see the translations in the table 2. As we see, it seems to be that the model performs considerably well on the short sentences but becomes poor in the longer sentences. During observation it is found that not all the numbers are translated right. Some numbers, to which the model does not know to translate(like 9) are by default represented as number 1. Also, when the rare words are encountered, the prior word tends to be repeated for quite a few times. When tested on a new common sentence of our choice, the performance is poor showing its not so good generalization ability. These are the few drawbacks of the model that has been created. Although it did have a few setbacks these could be made better by using a parallel corpus that is more general to normal conversation usage and by inculcating few model tuning practices. This could be by training for more iterations, by using corpus with huge vocabulary size, by adapting an optimizer that could generalize well (SGD), by increasing the number of layers in the encoder and decoder, by increasing the hidden dimension of the model etc. For simplicity purposes, in this implementation, PHP parallel corpus is used. Also, the OOV (Out of vocabulary words) which are rare words also matter as they could become one of the potential causes for performance degradation.

Table 1: Actual Vs Translated Sentences

Ground Truth	Translated Sentence
1) msql_fieldtype() is similar to the msql_fieldname() function.	msql_field_field() is used to msql_field()
2) number of messages	number of messages
This extension has no constants defined.	This extension has no configuration directives defined in php.ini.
3) If the link identifier isn 't specified, the last opened link is assumed.	If the link identifier isn' t specified, the last opened link is assumed.
4) This function used to be called pg_locreate().	This function used to be called pg_lo_lo_lo_lo_lo_lo_......
5) Example 1. bzdecompress()	Example 1. bzread() Example
6) The time is returned as a Unix timestamp.	The time is returned as a timestamp
7) Example 2. fbsql_query() example	Example 1. fbsql_query() example
8) Appendix D.	Appendix C.
9) Example 31-2.	Table 5-1.
10) If length is given and is positive, then the sequence will have that many elements in it.	The length is specified by length.
11) This example would display:	This will produce:
12) We cannot recommend any of those, as they tend to become out-of-date very quickly.	These functions are not because it is being called if you can be set.

5 Conclusion

In this implementation, we recreate the original transformers work of 2017 “Attention is all you

Encoder Layers	6
Decoder Layers	6
Hidden Dimension	512
Encoder Heads	8
Decoder Heads	8
Learning Rate	1.00E-04
Dropout	0.3
Loss Function	Negative Log Likelihood
Optimizer	Adam
Length of sentence	200
Source Vocab	13523
Target Vocab	9811

Table 2: Best Hyperparameters

need". We take a practical look at how the model is orchestrated and built to replace the traditional RNNs that we were using for machine translations. This is a breakthrough work that explains the importance of the context using the attention mechanisms which were not implemented so far. Transformers are good flexible architectures which could be used in a wide range of applications regardless of the application domain such as Language processing or Computer Vision. Although they have few limitations like context fragmentation due to chunking of the input sentence and ability to work with fixed length of text string, they have been overcome by the successor models. It is their perks that urges people to use them to their best ability. They are fast at processing a sequence since they pay more attention to its most important parts which makes it a best fit for the task. These are not only used for seq-2-seq language translations but also have various types such as ViT, a Vision transformer which works with images where the input image is broken down into a 16x16 grid of patches. They are also successfully used in Object Detection (DETR). All of them achieve state-of-the-art results and outperform current industry standards in the tasks. So, I conclude that this implementation paved a way to endless possibilities of implementations and their improvements.

6 Acknowledgement

I would like to express my deep gratitude to Dr. Jingyi Zhang, my Project supervisor, for her patient guidance, enthusiastic encouragement and useful critiques of this work. I would also like to express my very great appreciation for her valuable and constructive suggestions during the planning and development of this Project. Her willingness to give her time so generously to advice and assist in keeping my progress on schedule has been very much appreciated.

References

Alex Sherstinsky. 2018. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *CoRR*, abs/1808.03314.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

[1](Vaswani et al., 2017)

(Sherstinsky, 2018)

Code reference :- https://github.com/jadore801120/attention-is-all-you-need-pytorch