Extremely Low Resource Neural Machine Translation from Sanskrit to English

Vijaya Teja Rayavarapu
rayavarapu.v@northeastern.edu

Nikita Salkar
salkar.n@northeastern.edu

Nissar Ahmed
pinjari.n@northeastern.edu

Shreeyash Yende
yende.s@northeastern.edu

Khoury College of Computer Science, Northeastern University
Boston, MA

Abstract

Neural Machine Translation in a low-resource setting has been an obstacle yet to be overcome. Due to the sequential nature of the translation process, recurrent neural networks and long short-term memory networks have been used in the past. Although the Transformer architecture is a rising favorite for the task, its effectiveness has been tested only on large sets of language pairs. In this paper, we investigate the efficacy of the transformers in an extremely low-resource environment. Our results indicate that fine-tuning the hyperparameters is imperative in order to achieve an optimal transformer. With a dataset of 38 thousand sentence pairs of English and Sanskrit, our optimized transformer achieves a near state-of-the-art result with a BLEU score of 9.119.

Introduction

Ever since the Transformer architecture was introduced in 2017, it has made promising impacts on various applications including neural Machine Translation. Researchers have adopted specific configurations that enables it to produce optimal results. However, the transformer architecture has been largely successful only with large-scale Machine Translation datasets. Its usability and efficiency in a low-resource setting with limited amount of parallel corpora is still a question that remains unanswered. [9] and [10] present work in a similar setting and observe that in order for the transformer models to produce substantial results, extreme efforts are required to carefully experiment with and fine tune the hyperparameters to get the desirable results. In our study, we present work in a similar setting of an extremely low resource environment. Having a comparable dataset size, while [10] achieves a BLEU score of 7.3 (on a different language but with a scare dataset), our attentively chosen hyper-parameterized transformer achieves a BLEU score of 9.119.

To explain a little about the basic task at hand, Machine Translation (MT) is the task of automatically converting one natural language into another, preserving the meaning of the input text, and producing fluent text in the output language. It is one of the oldest subfields of Artificial Intelligence research. With the recent shift towards large-scale empirical techniques, there have been significant improvements in translation quality. Research over the years has led to broadly 4 types of Machine Translations.
Rule-Based Machine Translation (RBMT), Statistical Machine Translation (SMT), Hybrid Machine Translation (HMT), and Neural Machine Translation (NMT).

In Rule-Based Machine Translation, the system generates the translations of the input sentences based on morphological syntactic and semantic analysis of the 2 languages. Although this approach does not require bilingual data, it relies on good dictionaries and manually set rules.

Statistical Machine Translation uses bilingual text corpora along with phrase-based translations which translate sequences of words or phrases where the lengths may differ.

There are several types of statistical-based Machine Translation models which are: Hierarchical phrase-based translation, Syntax-based translation, Phrase-based translation, Word-based translation. However, it does not consider the context which often results in low-quality translations.

Hybrid Machine Translation is, as the name suggests, a mix of Statistical and Rule-based Machine Translation. Although its utilization of translation memory makes its output quality more successful, it requires a lot of editing and human translation. There are various approaches to HMT including multi-engine, statistical rule generation, multi-pass and confidence-based translation.

Neural Machine Translation also requires a bilingual corpus and uses evidently, neural networks to learn a statistical model for Machine Translation.

In this project, we use a neural Machine Translation model, namely the Transformer. The beauty of the Transformer architecture is that it processes the entire sentence in parallel during the training phase, hence reducing the training time. Transformers also entails the concept of self-attention and multi-headed attention wherein similarity scores between words within the same sentence are computed.

Low Resource Setting

Neural Machine Translation models feed on a large set of data to learn the nuances of the languages and achieve state-of-the-art results. Machine Translation models in the past, especially those that have surpassed human-level performance [2] have relied on the availability of large-scale parallel corpora; collections of sentences in both the source language and corresponding translations in the target language of around at least 2 million sentences.

However, in a realistic setting, the vast majority of languages don't have such resources. Facebook recently achieved a major milestone wherein they developed a novel approach that combines iterative back-translation and self-training with noisy channel decoding, to build an English-Burmese MT model [3].

Back translation effectively entails appending the output obtained, which is the translations of the source language, to the original dataset and iterating through this process again.

Another research conducted at MIT's CSAIL focused on automatic decipherment of lost languages in an ultimate low resource setting.[4]

Our project falls in the same category of being extremely low on resources since our exploration for data resulted in just about 38k parallel sentences.

Motivation

We had 3 main reasons for choosing Sanskrit to English Neural Machine Translation.

- After gathering our data, we wanted to evaluate the efficiency of transformers in such a low resource environment.
- Also, there are a vast number of texts written in Sanskrit that are yet to be translated for the common public. At the onset of our project, we envisioned building a state-of-the-art model that could translate these texts for them to be accessible by the common public.
- And our third driving factor was the fact that Google translate itself does not translate to or from Sanskrit.

Data Collection and Preprocessing

Datasets for Sanskrit to English translation were not readily available. There were no prior existing datasets such as the ones offered by WMT. So, we had to curate our own dataset. We crawled various websites in search of quality Sanskrit literature which had the corresponding English translations available.

After meticulously crawling the web, we scraped Holy texts which included The Ramayana [5], The Bhagavad Gita, The Rigveda, and The Bible along with their corresponding English translations. After scraping the web sites containing the above-mentioned holy texts, we then generated Sanskrit - English parallel sentence pairs using an automated script. The final size of our dataset spanned 38 thousand sentence pairs.

Below is a detailed summary of the statistics of our dataset.

Summary of Data Analysis

Description	Value
Total sentence pair count	37078
Total Sanskrit tokens count	849418
Total English tokens count	1108048
Maximum sentence length in Sanskrit	96
Maximum sentence length in English	104
Minimum sentence length in Sanskrit	4
Minimum sentence length in English	4
Mean sentence length in Sanskrit	22.908
Mean Sentence length in English	29.884
Maximum occurring subword count in Sanskrit	9698
Maximum occurring subword count in English	52681
Minimum occurring subword count in Sanskrit	1
Minimum occurring subword count in English	1
Train set size	33370
Test set size	3708

Table 1: Summary of Data Analysis

Following the synthesis of the data, we preprocessed the data to feed it to the Transformer.

The Main objective of data preprocessing is to convert the raw data obtained through various data resources into machine understandable format. It makes the knowledge discovery more efficient. This process plays a crucial in the training process as it prevents garbage data to be processed by the machine. We employed different data preprocessing steps starting with the removal of html tags and English content from the Sanskrit dataset. We then manually removed unnecessary introductions and other irrelevant lines from English dataset. These were various kinds of noise present in the dataset which we generated.

In order for the model to know when to begin and end the predictions, we added <bos> and <eos> tags to the beginning and end of the English translations. The Transformer implements Teacher Forcing by using a right shifted target sentence as reference and the actual target sentence as the input. Adding the <bos> and <eos> tags would provide this buffer at the beginning and the ending of the target sentence. We experimented with various sentence lengths to reduce the variance in the sentences and finally decided to restrict the length of the sentences to range from 3 to 50 for both the languages of Sanskrit and English.

Following this, the sentences had to converted to their byte -pair encoded sub word level tokens in order to reduce the occurrences of out of vocabulary words during the testing phase. For this process of converting the words in the sentences to their byte-pair encoded format we employed pretrained models provided by the BPEmb library for both English and Sanskrit languages. Once the sub-word tokens were ready, we created our own dictionaries for sub-word to indexes and vice-versa to fit our requirement rather than using the predefined dictionaries available with the tokenizers provided by BPEmb. Once these dictionaries were ready, for Sanskrit language we had 13486 sub-words and for English language we had 12662 sub-words.

Using the sub-word to index dictionaries we converted the sentences into their index representations. Once this was done, we sorted the sentence pairs based on combined sentence lengths in ascending order so that during the batching process we required minimal padding. Once the sorting was done, we created batches of sentence pairs by padding the shorter sentences based on the longest sentence.

Experiments

Representing the vocabulary: Owing to the size of our dataset, there were bound to be high instances of rare and out-of-vocabulary words. In order to tackle this, we used Byte-Pair Encoding for word segmentation. [8] suggests that decreasing the merge operations can yield better BLEU scores for recurrent NMT. We assumed it would work similarly for the Transformer architecture. After experimenting with different merge operations, we fixated the number of merge operations to 25000. We began our implementation by first getting an estimate of how well the Vanilla transformer performed. We trained our model with the following baseline parameters:

Number of Encoder/ Decoder layers	6/6
Number of heads	6
Embedding dimension	512
Feedforward Neurons	2048
Dropout	0.1
Smoothing	0.1
Batch size	256
Optimizer	Noam Optimizer with learning rate warmup of 4000

Table 2: Vanilla Transformer Parameters

The Vanilla model gave a baseline BLEU score of 6.855 with very inaccurate translations.

According to [6] the performance of the Transformer architecture can be enhanced by increasing the model parameters. However, as it is stated in [7], this is the case only in the ideal setting of a sufficiently large dataset and may not apply to a low-resource setting.

We explored values for several hyperparameters and tried various combinations of values. Since training the model by keeping each and every hyperparameter constant while experimenting with the rest would have been computationally heavy, we focused on the number of encoder/decoder layers, the feedforward dimensions, the embedding dimensions and the number of heads similar to [10]. The following table depicts the different hyperparameters explored and their corresponding values that were tried.

Sr	Hyper-Parameter Used	Values
1	Feed Forward Dimensions	256,512,1024,2048, 2187, 4096
2	Embedding Dimensions	128, 256, 512, 768
3	Encoder/Decoder Layers	1/1, 2/2, 3/3, 4/4, 5/5, 6/6, 7/7, 8/8, 2/5, 3/5, 3/4, 1/6
4	Smoothing	0.1, 0.3, 0.4, 0.5
5	Dropout	0.1, 0.2, 0.25, 0.3, 0.35,
6	Merge Operations for BPE	5000, 10000, 25000
7	Optimizers	Adam, AdamW ASGD, SGD, AdaGrad, AdaDelta, RMSProp
8	Activation Functions	ReLU, GELU, SiLU
9	Learning Rate Scheduler	Noam, LRonPlateau, No Scheduler
10	Batch sizes	1,4,8,10,15,16,20,32 ,64,128,256
11	Loss functions	KLDivergence, sparse_categorical_cross_entropy
12	Decoder	Greedy Decoder

Table 3: Hyperparameters and corresponding values explored.

Results

After experimenting with different combinations of hyperparameters, and comparing the BLEU scores of the models obtained, we got 2 models that resulted in exceptionally high BLEU scores of 9.119 and 8.855.

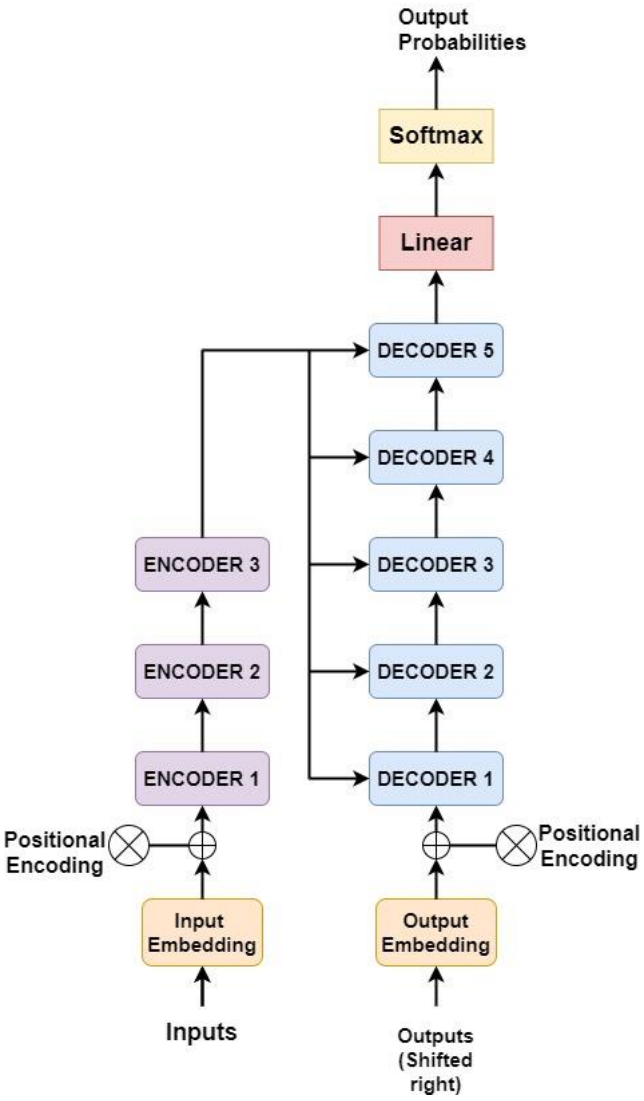
Our top 2 models had the following parameters:

	Model 1	Model 2
Feedforward Dimension	1024	512
Embedding Dimension	256	512
Number of heads	2	4
Encoder layers	3	3
Decoder layers	5	5
Encoder dropout	0.0	0.0
Decoder dropout	0.2	0.2
Positional Encoding dropout	0.2	0.2
Model dropout	0.3	0.3
Learning rate Scheduler	Noam	Noam
Optimizer	Adam	Adam
Smoothing	0.4	0.4
Batch size	128	128
BLEU score	9.119	8.855

Table 4: Hyper-parameters and findings of our models.

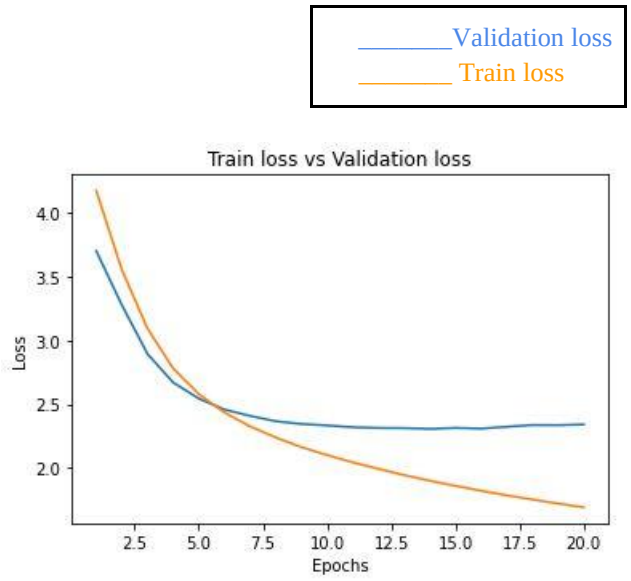
An interesting observation that can be inferred from both the models is that the Encoder and Decoder layers are not the same unlike as given in (Vaswani et. al.) [1]. Both the models have a total of 3 encoder layers and 5 decoder layers in contrast to 6 encoder and 6 decoder layers in the original transformer.

The architecture of our tuned transformer can be visualized as follows:



The individual encoder and decoder constituents remain the same as defined in (Vaswani et. al.) [1].

We depict the train and validation loss after 20 epochs. While the training loss decreased till the last epoch, the validation loss was observed to begin climbing after the 20th epoch. However, for the size of the data, the behavior of the model seemed reasonable. The training and validation losses of the Model 1 can be visualized as follows:



The translations obtained from the best model - model 1, are as follows:

1. **Translated**: thy spirit that went far away to the sea and went far away.
   Ground Truth: thy spirit that went far away away unto the billowy sea.

2. **Translated**: hearing the words of rama, the son of wind god, hanuma the son of wind god spoke to rama as follows:
   Ground Truth: hearing the words of rama, hanuma the son of wind god, who was excellent in the art of expression, forthwith spoke once more to rama as follows

3. **Translated**: for thee a new hymn lord of bays is fashioned may we be victors evermore be victors.
   Ground Truth: for thee a new hymn lord of bays is fashioned may we carborne through song be victors ever

The translations from model 2 are as follows:

1. **Translated**: may he with whose counsel my elder brother has gone to exile, incur the sin of killing a son, with whose counsel my elder brother has gone to exile.
   Ground Truth: may he with whose counsel my elder brother has gone to exile, possess many dependents be without resources, with fever and disease and be forever in distress.

2. **Translated**: indra and agni the aśvins ye gods preserve us evermore with blessings.
   Ground Truth: indra and agni and the aśvins lauded preserve us evermore ye gods with blessings

Conclusion

Research pertaining to Neural Machine Translation from Sanskrit to English using Transformers is, to our knowledge, non-existent as of now. Thus, with unprecedented work on the same, we sought to assay the performance of the Transformer architecture after fine-tuning the hyperparameters on our use case.
Given the challenges we faced which included

- An extremely low resource dataset
- Inconsistent translation style
- High variance in the target language and,
- No previous work for guidance,

our model achieves a near state-of-the-art performance with a BLEU score of **9.119** with very little training time. We believe that given more data, the accuracy of our model can be improved. Even with the current amount of data, there is still room for improvement by further fine-tuning and further careful selection of the hyperparameters.

Future Work

The efficiency of translation models may be improved by implementing the following:

- Back translation techniques can be used to generate synthetic data that can aid in increasing the dataset.
- Experimenting with Beam Decoder to see how the translation quality would be altered, is also something that can be worked upon.
- Like Sanskrit, there are a number of low-resource languages. Translation accuracy may get a boost by generating baseline models for all these low-resource distant language pairs.
- A language model can be implemented as a prior to the neural Machine Translation in order to improve the overall translation fluency of the model.

Contributions

All team members contributed equally to all areas of this project.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems (NeurIPS), Long Beach, CA, USA, 2017.

[2] CUBBITT - Transforming machine translation: a deep learning system reaches news translation quality comparable to human professionals by Martin Popel, Marketa Tomkova, Jakub Tomek, Łukasz Kaiser, Jakob Uszkoreit, Ondřej Bojar & Zdeněk Žabokrtský
https://www.nature.com/articles/s41467-020-18073-9

[3]https://ai.facebook.com/blog/recent-advances-in-low-resource-machine-translation/

[4]https://thenextweb.com/news/new-mit-algorithm-automatically-deciphers-lost-languages

[5] https://www.valmikiramayan.net/

[6] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. CoRR, abs/1910.10683

[7]https://arxiv.org/pdf/2011.02266.pdf

[8] Rico Sennrich and Biao Zhang. 2019. Revisiting low-resource neural Machine Translation: A case study. In Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, pages 211– 221.

[9] On Optimal Transformer Depth for Low Resource Language Translation
Elan van Biljon, Arnu Pretorius, Julia Kreutzer
https://arxiv.org/abs/2004.04418

[10] Optimizing Transformer for Low-Resource Neural Machine Translation
Ali Araabi, Christof Monz
https://arxiv.org/abs/2011.02266

[11] Extremely low-resource neural machine translation for Asian languages
Raphael Rubino, Benjamin Marie, Raj Dabre, Atushi Fujita, Masao Utiyama & Eiichiro Sumita
https://link.springer.com/article/10.1007/s10590-020-09258-6

[12] Revisiting Low-Resource Neural Machine Translation: A Case Study by Rico Sennrich, Biao Zhang
https://www.aclweb.org/anthology/P19-1021/
Volume: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics

[13] https://arxiv.org/abs/1508.04025
Effective Approaches to Attention-based Neural Machine Translation by Minh-Thang Luong, Hieu Pham, Christopher D. Manning

Appendix

We have uploaded the relevant code and results on github.

Link:
https://github.com/vijayaTejaRayavarapu/course_project_nlp/blob/feature/course_project_code.zip