Machine Translation with a Transformer

Apurba Das
40263612
apurba.das@mail.concordia.ca

December 11, 2024

Abstract. Google Translate or language translation features on social media platforms are now the norm. Behind these impressive technologies are sophisticated machine learning models that can understand and translate text between different languages. One of the most powerful and groundbreaking models used for this purpose is the transformer model. In this assignment, we built and trained our own transformer model for machine translation. We stepped into the shoes of an AI researcher and engineer to create our own Transformer model to translate text from English to French. This journey enhanced our understanding of machine learning and deep learning and also gave us hands-on experience with state-of-the-art techniques in natural language processing. This report presents the implementation and experimentation of this transformer-based machine translation model for English-French translation. We provide an overview of the implementation process, the experimental methodologies, the results, and the insights drawn from the analysis, along with playing around with a hyperparameter of our choice to measure its effect on the translation, and comparing the results of our model with the performance of using the T5 pre-trained model. The limitations of the study and potential improvements are also discussed.

Contents

1 Introduction

In this assignment, we develop and train a transformer model for English-French translation. The focus was on understanding the architecture, implementing the model and evaluating its performance using experimental data. Key findings include the effectiveness of hyperparameter tuning and the impact of dataset size on translation accuracy.

2 Experiments

The experiments involved building the transformer model [3, 2], training it on a bilingual dataset, and evaluating its translation capabilities. In the following, we provide detailed accounts of two experiments conducted, along with their limitations.

2.1 Experiment 1

The first experiment involved hyperparameter tuning with different values to measure its effect on translation.

2.1.1 Motivation

As part of our initial training, we trained our transformer model using google-bert/bert-base-multilingual-uncased tokenizer. We started with a learning rate of 0.0001 on training subset of 100000. We were able to train on 2.5 epochs when the gpu runtime got disconnected.

As the screenshot (figure 1 in appendix) shows, as the number of epochs were increasing, the train and validation loss kept decreasing indicating that we were on the right path. As we had saved this model, we proceeded with testing this on the entire testing set. We implemented greedy decode as seen in class in the NLG slides.

The model worked on a simple sentence (although the accuracy was not that good). Refer figure 2 of appendix. We tried running with complete test_set, but after running for a long time, the runtime got disconnected and we did not get any output. GPU was not getting reconnected post that due to collab limitations. Hence, we reduced the test set size to 10 and tried running it to get results with CPU. But we did not get any desired output (figure 3 in appendix).

By this time, Alejandra (the POD) posted on slack suggesting we try using FacebookAI/xlm-roberta-base as the tokenizer. This could be used as an alternative to m-bert or for comparison. Thus, we took her suggestion and that was our main motivation for this experiment. We hoped to get better results with the new tokenizer, and experiment with different parameter to see which produced best results with our limited gpu availability.

2.1.2 Methodology

The methodology involves several steps to train and evaluate the model effectively:

1. Data Preprocessing

- Extract translations for English (`en`) and French (`fr`).
- Standard preprocessing: converting text to lowercase, removing punctuation, numbers, and special characters.
- Tokenize text using the `FacebookAI/xlm-roberta-base` tokenizer with padding and truncation (maximum length: 120 tokens).

2. Model Design

- Implemented a Transformer model with:
  - **Embedding layers** for source and target languages.
  - **Positional encoding** for word order representation.
  - **Transformer architecture**: 512-dimensional embeddings, 8 attention heads, 3 encoder/decoder layers, and 256 feedforward dimensions.
  - A **linear layer** to map output to vocabulary space.

3. Training Process

- Optimizer: Adam with learning rate varied from 0.01, 0.001 and 0.0001.
- Loss function: Cross-Entropy loss, ignoring padding tokens.
- Masks:

  - **Padding masks** for handling input padding.
  - **Triangular masks** to prevent decoder from looking ahead.

- Training:
  - **Training set size** comparison was done with 10000 and 50000
  - **No. of training epochs** varied from 1 to 7

3. Evaluation

- Use validation and test sets to evaluate the model.
- Implement greedy decoding for translation, where tokens are predicted sequentially until an end-of-sequence token is reached.
- Compute metrics using BERTScore (Precision, Recall, F1) and METEOR for further evaluation.

2.1.3 Results

Refer to figure 5 in appendix.

2.1.4 Analysis

With a learning rate of 0.01 and training set of 10000 with 1 epoch we did not get any results. The POD informed that transformers model are highly sensitive to change and an increase of the learning rate to 0.01 can have an adverse impact. We then tried with 0.001 where the simple sentence got translated with repeated tokens and produced 0 on all the evaluation metrics. On changing the learning rate to 0.0001 with 2 epochs, we now got two words repeating but the translation of the word 'you' was correct. We then decided to stick to 0.0001 as the learning rate, and play around with the number of epochs and training and test set size. On increasing the training set size to 50000, we got better performace metrics of 0.00888, 0.00897, 0.00893, 0.0 for Precision, Recall, F1, Meteor respectively, and on increasing the number of epochs to 7 (5 more epochs on top of the 2 already done), we got a much better translation of the simple sentence (figure 4 in the appendix), with evaluation metrics as (0.8695, 0.8710, 0.8699, 0.5976).

2.1.5 Limitations

Due to GPU and memory resource constraints, we were not able to train on the entire training data, with much higher batch size and many more number of epochs. However, what we have done gives us a fair idea of which direction would be best suited to proceed for our model training.

2.2 Experiment 2

Experiment 2 involved running with t5 pretrained model. [1]

2.2.1 Motivation

The motivation for this experiment is to evaluate the performance of the T5 model on our specific natural language processing task of English to French translation. T5 (Text-to-Text Transfer Transformer) provides a unified framework for handling diverse NLP tasks by converting all problems into a text-to-text format. This experiment aims to understand its efficacy in terms of accuracy, generalization, and computational efficiency.

2.2.2 Methodology

The methodology involves the following steps:

1. Data Preparation: A dataset consisting of input-output text pairs was curated and preprocessed to match the T5 input format.

2. Model Training: The T5-small variant was fine-tuned on the dataset using a learning rate of $3e-5$ and a batch size of 8 for 5 epochs.

2.2.3 Results

Refer to figure 6 in appendix

2.2.4 Analysis

The T5 model demonstrated robust performance across diverse metrics. Qualitative evaluation showed that the model generated coherent and contextually appropriate responses.

2.2.5 Limitations

Again, due to resource constraints, we could not train with the entire dataset or with much higher number of epochs. However, this model was much more optimal both in terms of training time and performance results as compared to our previous model.

3 Conclusion and Future Work

3.1 Summary of your work and your findings

This experiment analyzed the custom transformer (with m-bert tokensiser (CLS and SEP) and XLM-R (BOS and EOS)) and T5 model's performance on our given text translation task. The experiment underscores T5's versatility in handling diverse NLP tasks in a unified framework. The findings suggest that T5 can effectively generate semantically and syntactically accurate text, supported by high scores across all evaluation metrics as compared to a custom built transformer model with m-bert and xlm-r tokenizer which requires much more training time and computational resources while still producing results which are subpar compared to T5.

3.2 Limitations

Despite promising results, the experiment faced several limitations:

- Limited exploration of larger model variants due to computational constraints.
- Lack of extensive hyperparameter tuning, potentially affecting performance.

3.3 Future Work

If provided with sufficient resources, future work could focus on:

- Exploring larger T5 variants (e.g., T5-base, T5-large) to assess scalability and performance improvements.
- Incorporating training and testing with the entire given dataset to evaluate the model's robustness, and trying out other similar datasets too to increase the range and diversity of vocabulary learnt.
- Experimenting with advanced optimization techniques to enhance fine-tuning efficiency.

References

[1] Translation. https://huggingface.co/docs/transformers/en/tasks/translation, 2024.

[2] Jurafsky, D., and Martin, J. H. Ch 9 - the transformer. https://web.stanford.edu/~jurafsky/slp3/9.pdf, 2024.

[3] Kosseim, L. Comp 6781 transformers and llms. https://moodle.concordia.ca/moodle/pluginfile.php/7197529/mod_resource/content/1/COMP_678 Transformer-annotated.pdf, 2024.

A Appendix

for tables, graphs and figures only

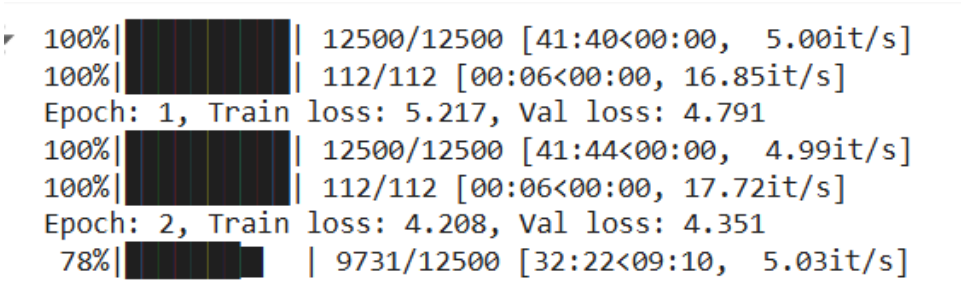

Figure 1: Training on multilingual bert

```
print(translate(model, "Hello how are you today",tokenizer))
```

```
bon comment sont aujourdhui aujourdhuissss
```

Figure 2: Simple sentence translation on multilingual bert

```
# Run the test function
test(test_set[:10], model, tokenizer, device)
```

```
(0.0, 0.0, 0.0, 0.0)
```

Figure 3: Test results on multilingual bert

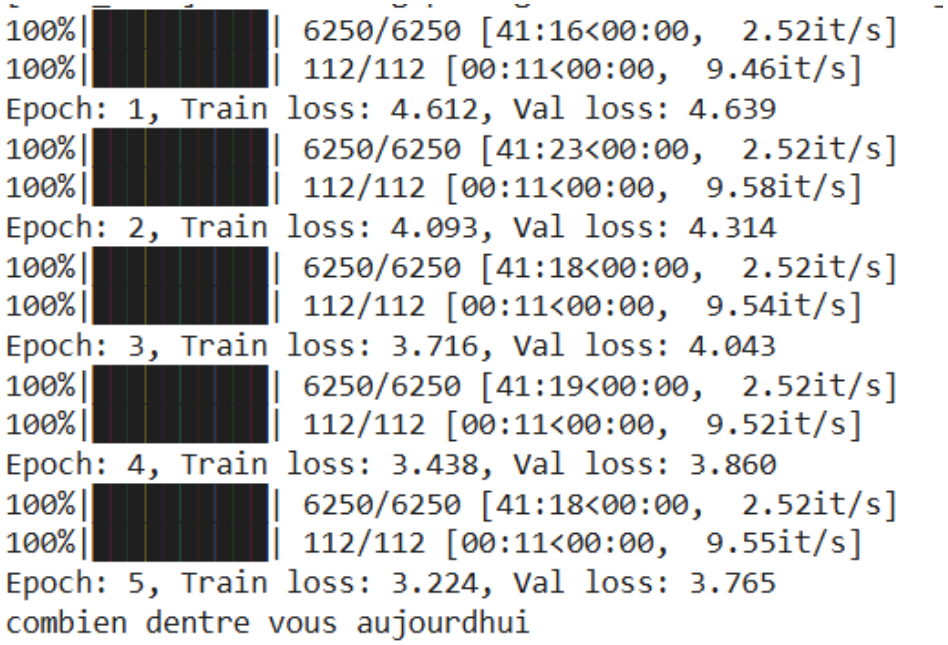

Figure 4: Hyperparameter tuning best result sentence translation along with epochs and loss during training

Model tokenizer	Learning rate	No. of training epochs completed	Training set size	Test set size	Sentence translation (Hello how are you today)	Results (Precision, Recall, F1, Meteor)
google-bert/bert-base-multilingual-uncased	0.0001	2.5	100000	Complete	Bon comment sont aujourd'hui aujourdhuissss	GPU disconnected before producing results
				100		GPU disconnected, produced warning
				10		0,0,0,0
FacebookAI/xlm-roberta-base	0.01	1	10000	10	No result	No result
	0.001	2	10000	10	Et et et et et	0,0,0,0
	0.0001	2	10000	10	Vous savez vous savez vous savez	0,0,0,0
	0.0001	2	50000	100	comment	(0.00888, 0.00897, 0.00893, 0.0)
	0.0001	5+2=7	50000	100	combien dentre vous aujourdhui	(0.00889, 0.00908, 0.00898, 0.000335)

Figure 5: Hyperparameter tuning results

Model	Learning rate	No. of training epochs completed	Training set size	Test set size	Sentence translation (Hello how are you today)	Results (Precision, Recall, F1, Meteor)
T5	3e – 5	5	50000	100	Bonjour, comment êtes-vous aujourd'hui	(0.8695, 0.8710, 0.8699, 0.5976)

Figure 6: T5 results