Transformer Based Machine Translation From English To Mandarin

Jiayuan Shen
MPML
Email: jshen20@umd.edu
UID: 118464947

Qiao Qin
MPML
Email: qqin@umd.edu
UID: 120409875

Ke Xu
MPML
Email: kxu233@umd.edu
UID: 120161216

I. Introduction

In this project, we implemented a transformer-based architecture to develop a machine translation model that translates English sentences into Mandarin. Our objective was to achieve translations that are comprehensible at a human level, maintaining the semantic integrity of the original English inputs.

The model employs a supervised learning approach, utilizing external datasets comprising pairs of English sentences and their corresponding Mandarin translations, which have been manually annotated. This data facilitates the training of our model to accurately capture linguistic nuances between the two languages.

Specifically, we utilized an encoder-decoder transformer architecture. This approach leverages the transformer's ability to handle sequential data and its effectiveness in learning contextual relationships within and between languages.

II. Dataset and Preprocessing

A. Dataset

The dataset was downloaded from hugging face: 'magicsword/train-en-zh'. It contains 100,000 English sentences and their corresponding Chinese sentences.

Here is an example of one sample in the dataset:
I have big news, everybody.|| 各位，我有重大新闻要宣布。

So first of all we will separate this sample into two columns: one is the English sentence and the other one is its corresponding Chinese translation.

B. Preprocessing

During the preprocessing, we use two tokenizers that are downloaded from hugging face for English and Chinese: 'bert-base-cased' and 'google-bert/bert-base-chinese'.

After tokenization, the results is:
['[CLS]', 'I', 'have', 'big', 'news', ',', 'everybody', '.', '[SEP]’]
['[CLS]', '各', '位', '，', '我', '有', '重', '大', '新', '闻', '要', '宣', '布', '。', '[SEP]']

It shows that the English sentence is tokenized at the word level, while the Chinese sentence is tokenized at the character level. Additionally, we add [CLS] and [SEP] as special tokens to indicate the beginning and end of each sentence. Note that [CLS] and [SEP] tokens are not originally used to indicate sentence boundaries, but since we are training a model from scratch and the model has not seen any tokens before, we simplify the usage of [CLS] and [SEP] tokens when using a BERT-based tokenizer.

Then, we convert these English and Chinese tokens into integers, mapping each token to its corresponding ID in the vocabulary:
[101, 146, 1138, 1992, 2371, 117, 10565, 119, 102]
[101, 1392, 855, 8024, 2769, 3300, 7028, 1920, 3173, 7319, 6206, 2146, 2357, 511, 102]

Finally, we save the entire dataset into a JSON file, making it convenient to directly access the preprocessed dataset for each training session.

III. Model and Methodology

A. Model Architecture

Our model architecture utilizes the standard Transformer as demonstrated in the ”Attention is All You Need” paper, and it invokes the torch.nn.Transformer class from PyTorch. The specific parameter settings of the model are shown below.

TABLE I
Model Parameters

Parameter	Value
Number of Blocks	N=3
Embedding Size	$d_model = 512$
Feed Forward Dimension	$d_{ff} = 1024$
Number of Heads	h=8
Heads Size	$d_k = d_v = 64$
Dropout Probability	$P_{drop} = 0.1$

Our model parameters’ size is slightly less than the base model in original paper due to the limitation of computational resources.

B. Methodology

1) Padding Mask: Batching involves packing multiple samples into a single tensor. However, to create a valid tensor, each sample's length should be the same, which is not the case as each sentence has a different length. Therefore, we need to pad the sentences using a special token [PAD] (corresponding to ID 0). However, since [PAD] itself does not affect the meaning of the sentence, we need to instruct the model not to pay attention to it. Thus, we need to pass a padding mask to the model to indicate which part of the sentence is padded.

As an example, assuming a batch size of 3, before padding, the three sentences look like this.

Sentence 1 : [101, 146, 1138, 1992, 10565, 119, 102]

Sentence 2 : [101, 1252, 170, 2113, 111, 1139, 5080, 119, 102]

Sentence 3 : [101, 2353, 1653, 117, 102]

After padding, zeros are added to the end of each sentence to match the length of the longest sentence.

Sentence 1 : [101, 146, 1138, 1992, 10565, 119, 102, 0, 0]

Sentence 2 : [101, 1252, 170, 2113, 111, 1139, 5080, 119, 102]

Sentence 3 : [101, 2353, 1653, 117, 102, 0, 0, 0, 0]

Then, a mask containing only 0s and 1s is used to indicate which parts are [PAD].

Sentence 1 : [0, 0, 0, 0, 0, 0, 0, 1, 1]

Sentence 2 : [0, 0, 0, 0, 0, 0, 0, 0, 0]

Sentence 3 : [0, 0, 0, 0, 0, 1, 1, 1, 1]

2) Attention Mask: Attention masks are a vital component of the machine translation model based on the Transformer architecture. These masks control which parts of the input sequence the model should focus on during decoding.

In machine translation, the decoder generates each word of the translated sentence one at a time, it would be cheating to see the answer from future. By using attention masks, we ensure that the decoder only attends to current and previous part of the sequence, considering only the tokens that have been generated so far.

Specifically, the attention mask is a T by T lower triangular matrix, where T is the length of the sequence. Elements on the main diagonal and below are set to 1, while elements above the main diagonal are set to 0. This mask, enables the model to mask out future parts of the sequence.

3) Teacher Forcing: Teacher forcing involves feeding the model with the ground-truth target sequence during training instead of its own predictions, which is different from auto-regressive, and we will take about it next. Teacher forcing means that at each time step, the model receives the correct next token as input, regardless of its own previous predictions. This technique helps stabilize and accelerate the training process by providing more accurate signals to the model during training. It also mitigates the issue of exposure bias, where the model may struggle to generate accurate predictions during inference if it has only been exposed to its own imperfect predictions during training.

4) Auto-regressive: The model generates each token in the translated sequence sequentially, conditioning its predictions on previously generated tokens. At each time step, the model uses its own predictions as input to generate the next token in the sequence. This approach allows the model to leverage context from the already generated tokens, enabling it to produce fluent and coherent translations. For the sake of simplicity, we do not apply beam search, but use greedy search, where the token with the highest probability is chosen as the prediction.

IV. Training and Evaluation

A. Training

Some hyperparameters we use for training, including the learning rate, the number of epochs, the batch size, the optimizer and how we control the learning rate decay, are shown in the table below.

TABLE II
Training Hyperparameters

Hyperparameter	Value
Learning rate	8e-5
Optimizer	AdamW
Learning rate scheduler	CosineAnnealingLR
Number of epochs	15
Batch size	128

B. Evaluation

In this project, we use two metrics to evaluate our model: BLEU score and BERT score.

BLEU score is a widely-used metric for evaluating the quality of machine-translated text against one or more reference translations. It measures the similarity between the generated translation and the reference translations based on n-gram precision. BLEU score ranges from 0 to 1, with higher scores indicating better translation quality. It provides a measure of how well the generated translation aligns with human judgments.

BERT score is a more recent evaluation metric that leverages pre-trained BERT models to compute similarity between generated and reference translations. Unlike BLEU score, BERT score considers semantic similarity by comparing embeddings of words and phrases in the generated and reference translations. This makes BERT score more robust to syntactic variations and word order differences.

The evaluation based on the two metrics before and after training are shown in Fig I. It can be seen that after 15 epochs of training, the model's performance has significantly improved according to various evaluation metrics. However, this level of accuracy is still insufficient. We will provide detailed explanations of specific limitations in the subsequent section on limitations.

Here are the translation results of the model before training. As we can see, the translation results at this point do not make sense at all.

Input: ’How old are you ?’

Output: [CLS] 条条条…

And here are the translation results of the model after training.

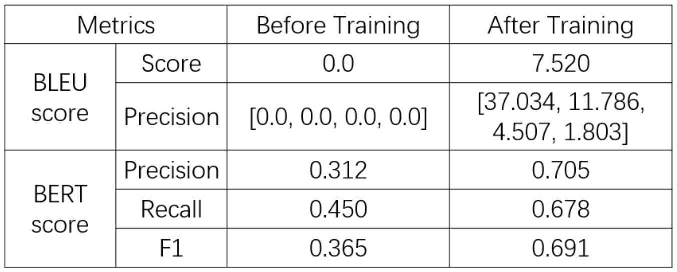
Input: ’How old are you ?’

Output: [CLS] 你几岁？[SEP]

Input: ’Are you worried about the African front ?’

Output: [CLS] 你还有担心吗？[SEP]

The first example comes from the training set, and the correct translation of this Chinese sentence is: ” Are you



Metrics		Before Training	After Training
BLEU score	Score	0.0	7.520
	Precision	[0.0, 0.0, 0.0, 0.0]	[37.034, 11.786, 4.507, 1.803]
BERT score	Precision	0.312	0.705
	Recall	0.450	0.678
	F1	0.365	0.691

Fig. 1. Evaluation

worried about?” The second example is an arbitrary input, and the correct translation of this Chinese sentence is: ”How old are you?” As you can see, although the translation results are still not very accurate, they are starting to make some sense.

V. Limitation

The main drawback of our model is its low accuracy, as demonstrated earlier, with a BLEU score of only 4.402. For comparison, in the ”Attention is All You Need” paper, although their model is used for English to German translation, while ours is for English to Chinese translation, their base model achieved a BLEU score of 25.8. Furthermore, from the actual results, our model did not achieve overfitting on the training set. To improve this, possible directions for enhancement include: training the model for more epochs, increasing model complexity, and adjusting hyperparameters including learning rate.

Another significant drawback of the model is its slow translation generation speed because the model currently does not support multiple sentences input simultaneously. That is, during inference, it requires the batch size to be exact 1. This greatly slows down the evaluation speed. Due to this limitation, our evaluation was conducted only twice on the validation set, before and after training, instead of monitoring of the model ‘s performance on the validation set during the training process. This is the best we can do for now. To address this issue, a deeper understanding of the auto-regressive mechanism and corresponding code implementation are required.

VI. Conclusion

In this project, we successfully built a machine translation model from English to Chinese using a transformer-based architecture. Our efforts included acquiring and preprocessing the dataset, tokenizing the data, and converting it into a format suitable for the model. We then conducted supervised learning on a transformer encoder-decoder model using this dataset. During training, we implemented padding, generated attention and padding masks, and correctly applied teacher forcing. Additionally, during inference, we evaluated the model using BLEU and BERT score metrics and effectively implemented auto-regressive generation. These steps allowed us to achieve a functional and promising translation model, demonstrating the potential of transformer architectures in machine translation tasks.