COSE474 Deep Learning ASSIGNMENT4 REPORT

MuYeong Jeong
Korea University
2019320112
jmy3033@naver.com

Abstract

In this assignment, I conducted text classification using the "IMDb Movie Reviews" dataset with a Transformer-based model. The dataset contains 25,000 movie reviews for training and another 25,000 for testing. Each review is labeled as either positive or negative.

The Transformer is a neural network architecture designed for processing sequential data. Unlike traditional models, it handles the entire sequence at once and uses self-attention mechanisms to capture dependencies between tokens. Multi-head attention allows the model to focus on different parts of the sequence simultaneously.

1. Implementation

In this assignment, I implemented Positional Encoding, Multi-Head Attention, and the Transformer Encoder Layer. I will explain each component in order.

1.1. Positional encoding

$$PE(pos, 2i) = \sin\left(pos/(10000^{\frac{2i}{d\_model}})\right) \quad (1)$$

$$PE(pos, 2i+1) = \cos\left(pos/(10000^{\frac{2i}{d\_model}})\right) \quad (2)$$

Using the above formula, sine values were added to the even indices and cosine values to the odd indices. This allows the Transformer to utilize information about word order in the input sequence.

1.2. Multi-head Attention

$$scores = \frac{Q \cdot K^T}{\sqrt{d\_k}} \quad (1)$$

$$weights = softmax(scores) \quad (2)$$

$$output = dropout(weights) \cdot V \quad (3)$$

$$mutihead\ attention\ output = W_O \times output \quad (4)$$

The dot product between the query and key matrices was computed in parallel. Scaled scores were used because, without scaling, the dot product values can become too large, which may lead to the vanishing gradient problem during training. In addition, dropout was applied to prevent overfitting.

1.3. Transformer Encoder Layer

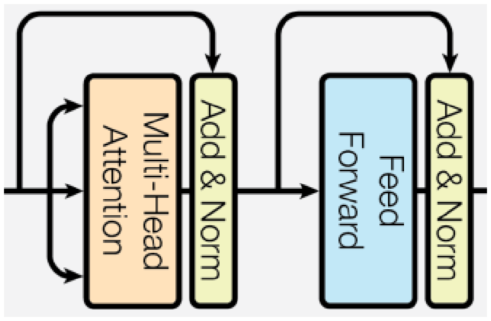

Figure 1: Layers of Transformer Encoder.

The layer was implemented based on the architecture shown in Figure 1. The formulas used are as follows.

$$temp1 = dropout(attention\ output) \quad (1)$$

$$output1 = normalization(input + temp1) \quad (2)$$

$$temp2 = dropout(feedforward\ output) \quad (3)$$

$$output2 = normalization(output1 + temp2) \quad (4)$$

2. Result

After training for one epoch, the model achieved a training loss of 0.6619 and a validation accuracy of 63.82%. The test accuracy was 64.28%.

After loading the pretrained weights from BERT, the model showed a significant improvement in performance. At epoch 1, the training loss was reduced to 0.4476, with a validation accuracy of 82.50%. The test accuracy also improved to 82.62%.

The Transformer model built from scratch starts learning without any prior knowledge, which leads to lower accuracy. In contrast, when using pretrained weights from BERT, the model already contains prior knowledge acquired through large-scale pretraining. Fine-tuning on the IMDb review classification task allows the model to adapt this knowledge effectively, resulting in a significant improvement in accuracy. The difference between the two models demonstrates the advantages of transfer learning.