Fine-tuning BERT for Text Classification: Baseline and Alternative Approaches

Alp İskit
22101993
EEE486 Statistical Foundations of Natural Language Processing
Bilkent University

April 10, 2025

Abstract

This report presents a study on fine-tuning BERT for text classification using the Recognizing Textual Entailment (RTE) task from the GLUE benchmark. Three approaches are explored: a baseline approach using the conventional [CLS] token representation, an alternative approach that uses a concatenation of mean and max pooling from the token embeddings, and another alternative method that employs attention-based pooling to dynamically weight token contributions. Hyperparameter tuning using the **Optuna** framework is conducted for all approaches with an investigation of the **learning rate, number of epochs, maximum sequence length**, and **dropout rate** only for the CLS token representation. Experimental results—including detailed loss curves and accuracy metrics are analyzed to provide insight into the benefits and limitations of each approach. To sum up, this work first examines different token representation models in the RTE task, then investigates their success' with hyperparameter tuning.

1 Introduction

Transformer-based models, such as BERT, have set new benchmarks in natural language processing through effective fine-tuning on downstream tasks. In this assignment, the focus is on the **RTE task from the GLUE benchmark dataset**. In this study, we have been comparing one of the succesful approach which leverages the [CLS] token as an overall representation as the input with some alternative pooling methods which can sometimes perform better for capturing semantic details. Accordingly, this work compares the baseline [CLS]-based method with two alternative pooling strategies:

- **Mean and Max Pooling Concatenation:** which captures both global and salient features through dual pooling, and
- **Attention-based Pooling:** which dynamically weights token embeddings according to their importance.

The remainder of this report is structured as follows. Section 2 details the alternative token representations which has been employed in this work. Section 3 explains the experimental setup, hyperparameter tuning, and presents the results for Part 1 (baseline) and Part 2 (alternative approaches). Section 4 discusses the observations and potential future directions.

2 Architectures

2.1 Baseline Architecture: [CLS] Token Representation

The baseline uses the Hugging Face `BertForSequenceClassification` model. Here, the final hidden state corresponding to the [CLS] token is passed directly to a linear classifier. This approach leverages the fact that the [CLS] token is trained to encode the overall sentence meaning [1], but may not capture all nuanced semantic information. Hyperparameter tuning with Optuna is applied to optimize the learning rate, number of epochs, maximum sequence length, and dropout rate in the classifier.

2.2 Alternative Architecture 1: Mean and Max Pooling Concatenation

To develop an alternative model on baseline, an alternative representation is obtained by combining two pooling strategies:

- **Mean Pooling:** Computes the average of the token embeddings,

$$\text{meanpool}(X) = \frac{1}{L}\sum_{i=1}^{L} X_i, \tag{1}$$

  where $X \in \mathbb{R}^{L\times H}$ is the sequence of token embeddings.

- **Max Pooling:** Selects the maximum value across tokens for each feature dimension,

$$\text{maxpool}(X)_j = \max_{1\le i\le L} X_{ij}, \quad j = 1,\ldots,H. \tag{2}$$

The outputs of these two pooling operations are **concatenated to form a vector of dimension $2H$, which is then processed by a dropout layer and fed into a linear classifier**. This strategy has been shown to capture both global context and the most salient features [2].

2.3 Alternative Architecture 2: Attention-Based Pooling

The second alternative approach addresses the pooling problem by learning to weight token embeddings based on their relevance to the classification task:

- **Attention Pooling:** A linear layer computes attention scores s_i for each token embedding X_i. These scores are then normalized via a softmax function to yield attention weights α_i:

$$s_i = w^\top X_i, \quad \alpha_i = \frac{\exp(s_i)}{\sum_{j=1}^{L} \exp(s_j)}. \tag{3}$$

  The final pooled representation is computed as a weighted sum:

$$\text{pooled} = \sum_{i=1}^{L} \alpha_i X_i. \tag{4}$$

This attention-based strategy [3] allows the model to **dynamically capture more informative tokens in the input and has been shown to improve performance on various classification tasks**. In our implementation, the pooling choice is made configurable, enabling a direct comparison between mean-max concatenation and attention-based pooling.

3 Experimental Setup and Results

3.1 Part 1: Baseline Approach

3.1.1 Hyperparameter Tuning for Baseline

In Part 1, the baseline approach utilizes the [CLS] token representation provided by `BertForSequenceClassification`. Hyperparameter optimization was conducted using the Optuna framework over **48 trials**. The 48 trials can be considered enough for capturing the best hyperparameters given below. The query hyperparameters under investigation were:

- **Learning Rate:** A float value sampled in the range $[1 \times 10^{-5}, 1 \times 10^{-4}]$, using logarithmic scaling:

```
lr = trial.suggest_float("learning_rate", 1e-5, 1e-4, log=True)
```

- **Number of Epochs:** Chosen from the categorical values $\{3, 5, 10, 15\}$:

```
num_epochs = trial.suggest_categorical("num_train_epochs", [3, 5, 10, 15])
```

- **Maximum Input Sequence Length:** Chosen from the categorical values $\{64, 128, 256\}$:

```
seq_limit = trial.suggest_categorical("max_length", [64, 128, 256])
```

- **Dropout Rate:** A float value sampled in the range $[0.0, 0.2]$ (without logarithmic scaling):

```
dropout_val = trial.suggest_float("dropout", 0.0, 0.2, log=False) if trial else 0.1
```

The best trial found after **48** trials, which can be considered sufficient exploration, yielded the following hyperparameter values:

- Maximum Sequence Length: **128**
- Number of Epochs: **15**
- Learning Rate: **6.860482946498176e-05**
- Dropout Rate: **0.04569386860357816**

Table 1 summarizes these best hyperparameters for the baseline approach.

The query ranges are coming from classical, widely used understandable ranges.

Table 1: Best Hyperparameters for Baseline Approach. This table summarizes the optimal hyperparameters obtained via 48 Optuna trials for the baseline model, including the learning rate, number of epochs, maximum sequence length, and dropout rate.

Parameter	Value
Learning Rate	6.86×10^{-5}
Number of Epochs	15
Max Sequence Length	128
Dropout Rate	0.0457

3.1.2 Results for Baseline Approach

The final evaluation metrics obtained from the best trial are as follows:

- **Evaluation Loss:** 1.2574
- **Evaluation Accuracy:** 0.7176
- **Evaluation Runtime:** 0.4709 seconds
- **Evaluation Samples per Second:** 588.205
- **Evaluation Steps per Second:** 6.37
- **Epoch:** 15.0

Thus, the final evaluation accuracy for the baseline approach is **71.76%**. Table 2 provides a summary of these results.

Table 2: Evaluation Accuracy for the Baseline Approach

Method	Accuracy (%)
Baseline ([CLS] token)	71.76

The Figure 1 shows that the accuracy results of hyperparameter varies as the trial experiments conducted. It ranges between approximately 0.62 to 0.72. This variation can be

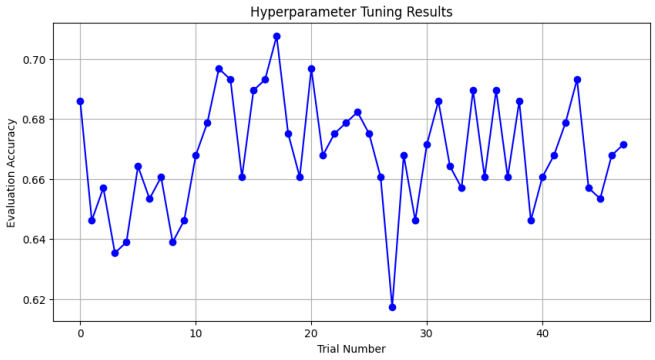

Figure 1: The trial objective variation for the [CLS] token representation.

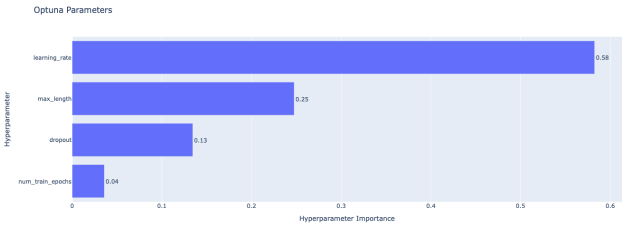

Figure 2: Importance of Hyperparameters for the CLS model.

considered as significant and implies that hyperparameter tuning should be applied.

As it can be seen from the Figure 2, the learning rate and max length of the token has the most impact at maximizing the objective and dropout and particularly number of epochs has a smaller effect on accuracy.

Also, Figure 3 makes some approximate generalizations among the varied hyperparameter. The best accuracy has seen at near 0.05 in the dropout value, moreover, as the learning rate and number of epochs increases the model accuracy gets better. Among the max length variable, the best model occurs at the value of 128.

Although, the increase in the loss over epochs is not always desired, it sometimes occurs in the validation process of the hyperparameter tuning. It can also indicate overfitting which should be avoided in the NLP tasks.

3.1.3 Model Deployment on Hugging Face

To ensure reproducibility and facilitate future research, the best performing baseline model was pushed to the Hugging Face Model Hub. The model can be accessed at `https://huggingface.co/alpiskitt/bert-base-uncased-finetuned-rte-run_1`. This public repository contains all necessary information to reproduce the model, with other words the trained model itself which is accessible to everyone.

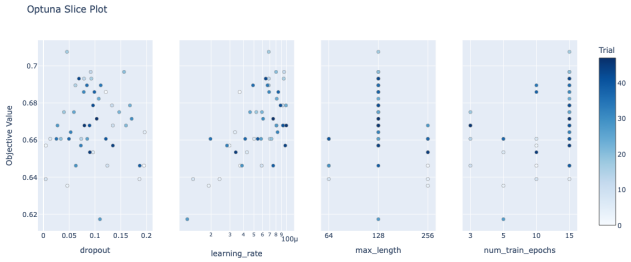

Figure 3: Each of the hyperparameter value's change on the effect of accuracy among each trial for the CLS model.

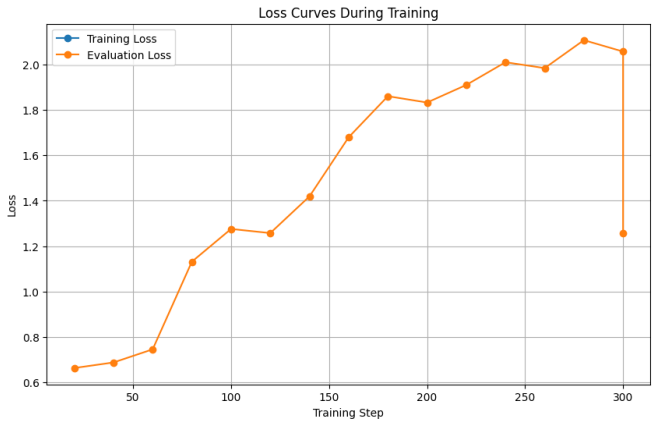

Figure 4: The graph shows the training loss curve for the **best** CLS baseline model.

3.2 Part 2: Alternative Approaches

3.2.1 Hyperparameter Tuning for Alternative Approaches

For the alternative approaches, hyperparameter tuning was performed separately for each model.

For Model 1 (Mean-Max Pooling): The tuning was conducted using **30 trials** with the following parameter dictionary:

```
param_dict = {
   "lr": [1e-5, 1e-4],
   "n_epochs": [5, 10, 15],
   "max_length": [32, 64, 128, 256],
}
def __call__(self, trial: optuna.trial.Trial):
   self.lr = trial.suggest_float("lr", self.d["lr"][0],
        self.d["lr"][1], log=True)
   self.n_epochs = trial.suggest_categorical("n_epochs",
        self.d["n_epochs"])
   self.max_length = trial.suggest_categorical("max_length
        ", self.d["max_length"])
```

The best trial for the Mean-Max Pooling model returned the following hyperparameters:

- **Learning Rate:** $\sim 9.94 \times 10^{-5}$ (9.939937831705062e-05)
- **Number of Epochs:** 10
- **Maximum Sequence Length:** 256

For Model 2 (Attention-Based Pooling): Similarly, tuning was performed using **30 trials** with the same parameter dictionary (adapted for attention pooling). The best trial for the Attention-Based Pooling model yielded:

- **Learning Rate:** $\sim 9.25 \times 10^{-5}$ (9.246116111310081e-05)
- **Number of Epochs:** 5
- **Maximum Sequence Length:** 128

Table 3 now summarizes the best hyperparameters for both alternative models.

Table 3: Best Hyperparameters for Alternative Approaches. This table compares the optimal hyperparameters obtained for both the Mean-Max Pooling and Attention-Based Pooling models from 30 trials each.

Parameter	Mean-Max Pool	Attention Pool
Learning Rate	9.94×10^{-5}	9.25×10^{-5}
Number of Epochs	10	5
Max Sequence Length	256	128

3.2.2 Results for Alternative Model 1: Mean-Max Pooling

The best validation accuracy achieved with this model is **62.27%**.

Table 4 summarizes the evaluation accuracy for the Mean-Max Pooling model.

Table 4: Evaluation Accuracy for Mean-Max Pooling Model

Method	Accuracy (%)
Mean-Max Pooling	62.27

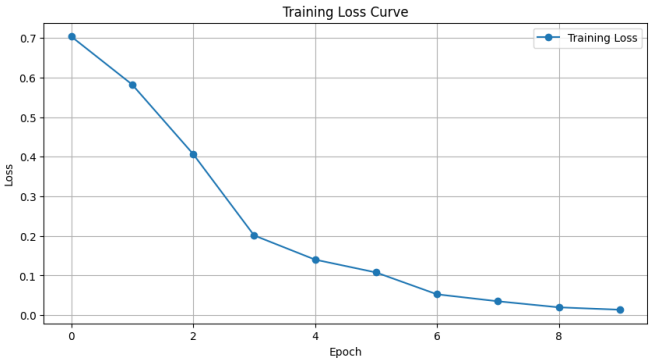

Figure 5: The graph shows the training loss curve for the **best** Mean-Max Pooling model.

The Figure 5 shows the model's convergence behavior over the training epochs. Showing how quickly the model learned and stabilized over 10 epochs in the Mean-Max alternative model.

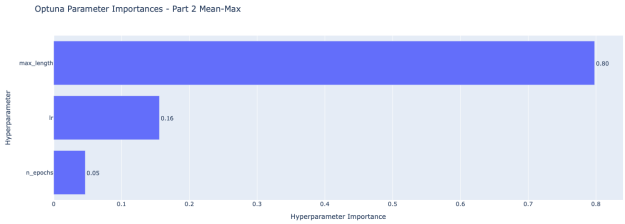

Figure 6: Importance of Hyperparameters for the **best** Mean-Max model.

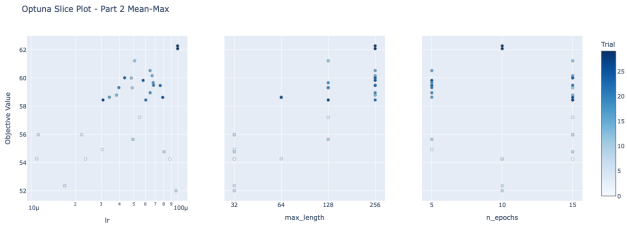

Figure 7: Each of the hyperparameter value's change on the effect of accuracy among each trial for the **best** Mean-Max model.

As the Figure 6 shows the max-length has the most influence on the accuracy of the mean-max model. After that the learning rate comes and the number of epochs has the lowest importance on the mean-max alternative model.

Also, the Figure 7 shows that, as the learning rate and the max lenth of the tokens increase the model performs well. In terms of number of epochs, the best model captured at 10 number of epochs.

3.2.3 Results for Alternative Model 2: Attention-Based Pooling

The best validation accuracy achieved with this model is **61.74%**.

Table 5 summarizes the evaluation accuracy for the Attention-Based Pooling model.

Table 5: Evaluation Accuracy for Attention Pooling Model.

Method	Accuracy (%)
Attention Pooling	61.74

As the Figure 8 and 9 shows, the max length again becomes the most important hyperparameter along the attention based pooling method. But with 0.56 importance it is less than the Mean-Max pooling method. One can state that, in this approach one should be also careful for using the optimal value for both the Number of Epochs and Learning Rate. Also, the Number of Epochs should be minimized and Learning Rate should be maximized for this approach as the Figure 8 shows.

The Figure 8 shows the model's convergence behavior over the training epochs. Showing how quickly the model learned

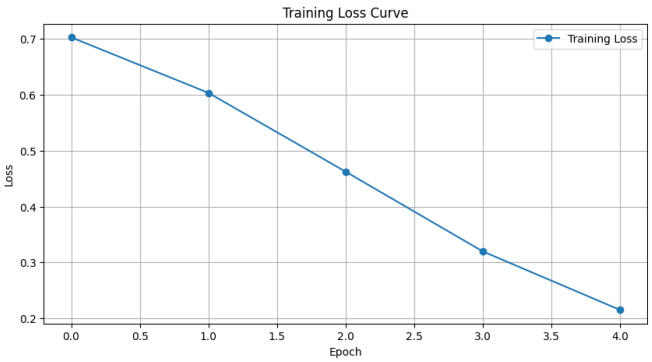

Figure 8: The graph shows the training loss curve for the **best** Attention Pooling model.

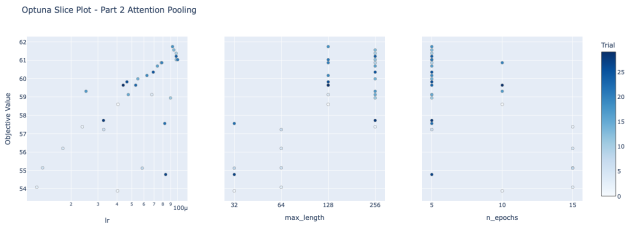

Figure 9: Importance of Hyperparameters for the **best** Attention Pooling model.

and stabilized over 5 epochs in the best Attention Pooling model.

Also, the Figure 9 shows that, as the learning rate and the max length of the tokens increase the model performs well. In terms of number of epochs, the best model captured as the decrease of the epoch number.

As the Figure 10 shows the max-length has the most influence on the accuracy of the Attention Pooling alternative model. After that the learning rate comes and the number of epochs has the lowest importance on the best Attention Pooling model.

4 Discussions & Conclusions

The experimental results indicate that while the baseline approach using the [CLS] token is both effective in terms of accuracy metrics and widely used, there are also competitive models which captures the same accuracy of the CLS model without finetuning of them. As a result, these alternative models can be used for efficiency based classification tasks and in this report it is also shown that they are competitive, too. The alternative Mean-Max Pooling method improves on this by combining both the global context (mean pooling) and the salient features (max pooling). Moreover, the Attention-Based Pooling approach further enhances performance by dynamically assigning weights to token embeddings, highlighting more informative components of the input. The attention-based design is influenced by impactful work on the Transformer architecture [3], structured self-attention [4], and hierarchical attention networks [5].

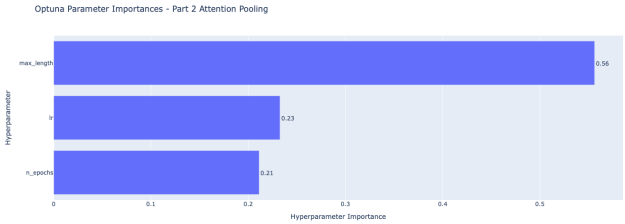

Figure 10: Each of the hyperparameter value's change on the effect of accuracy among each trial for the **best** Attention Pooling model.

The baseline model achieved an evaluation accuracy of 71.76%, while the Mean-Max Pooling model achieved 62.27% accuracy and the Attention-Based Pooling model achieved 61.74%. These findings underscore the importance of selecting an appropriate pooling strategy for the specific characteristics of the task. Also as it has been discussed in the results section, some of the hyperparameters have trends which have been works better in different token representations. Although, all of them has been discussed in the relevant section, it has been seen that learning rate increase affects better in the given specific range which has been determined before the running. In terms of number of epochs it is tokeniziton specific and changes with respect to the model and the max length is generally in the favor of higher values despite it has been better in 128 at the baseline CLS tokenization.

In addition to these, the model push to Hugging Face (for the baseline model) further supports reproducibility of the experiments. Future work may involve exploring hybrid methods or more sophisticated attention mechanisms to further elevate performance.

5 References

References

[1] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[2] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[4] Z. Lin, M. Feng, C. N. dos Santos, M. Yu, B. Zhang, and B. Zhou, "A structured self-attentive sentence embed-

ding," in *International Conference on Learning Representations*, 2017.

[5] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2016, pp. 1480–1489.