Fine-Tuning BERT on GLUE Benchmark

Athina Stewart
as1896@rit.edu
Rochester Institute of Technology
Rochester, New York, USA

Souvik Dey
sd5223@rit.edu
Rochester Institute of Technology
Rochester, New York, USA

1 TASK DEFINITION, EVALUATION PROTOCOL, AND DATA

1.1 The Natural Language Problem

Words are problematic in the sense that many are ambiguous, polysemous (having many meanings), and synonymous. Most words in the English language have multiple meanings and when it comes to spoken English specifically, the problem is made worse by homophones and prosody (the pattern of stress and intonation). A word has no meaning until use in a particular context and the meaning of a word can change as a sentence develops. For example in the sentence: "I like the way that looks like the other one", "like" has two meanings. Natural language understanding requires an understanding of context and common sense reasoning.

BERT (Bidirectional Encoder Representations from Transformers) was developed to address limitations in existing natural language processing (NLP) models. Traditional NLP models, especially those based on recurrent neural networks (RNNs) and convolutional neural networks (CNNs), had difficulty capturing the complex contextual relationships and nuances in language. BERT aimed to overcome these limitations and improve the representation of language by introducing bidirectional context understanding.

Traditional models processed text in a unidirectional manner (either left-to-right or right-to-left), which limited their ability to capture contextual information effectively. BERT introduced a novel bidirectional context understanding, allowing the model to consider both left and right context for each word, leading to better contextual representation.

In this report, we fine-tune a pre-trained BERT model and report the performance accuracy of fine tuning tasks on datasets found in the General Language Understanding Evaluation (GLUE) benchmark. This benchmark consists of nine sentence or sentence-pair language understanding (NLU) tasks built on existing datasets that are of varying sizes, genres and degrees of difficulty. The GLUE benchmark is a commonly used collection of resources for training, testing and analysing NLU systems. Of the entire benchmark set, we will be fine-tuning on both sentence pair tasks (MNLI, QQP, QNLI, MRPC and RTE) and single sentence classification tasks (SST-2 and CoLA).

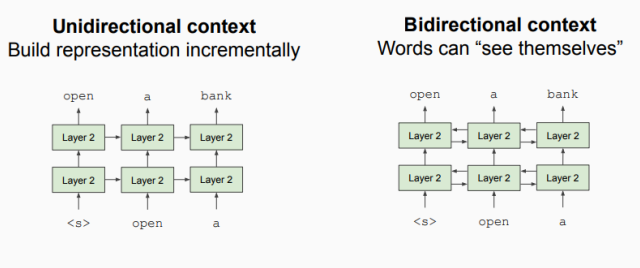

Figure 1: Unidirectional and Bidirectional Models[6]

1.2 Datasets

1.2.1 *MRPC.*

In 2005, the Microsoft research team created the Microsoft Research Paraphrase Corpus (MRPC). This corpus consists of 5,801 pairs of sentences. Each pair is rated by a binary human-delivered judgement of whether the pair of sentences is similar in meaning.[2]. This dataset was born out of the need for a large enough corpus for training Statistical Machine Translation (SMT) models for paraphrase tasks. An initial set of 13M sentence pairs were extracted from the web over a two-year period. A classifier was then built, with feature classes such as string similarity and WordNet Lexical mapping, skewed more towards positive and plausible "near misses" as the performance of the classifier itself

was not being evaluated. From the output, 5,801 pairs of sentences were randomly selected.

1.2.2 *MNLI*.

The MNLI dataset, created in 2018, aims to develop and assess machine learning models for sentence understanding. With an impressive 433k examples, MNLI is one of the largest language corpora. In contrast to SNLI, derived solely from image captions, MNLI addresses limitations by including diverse genres, capturing complexity in language. MNLI sources from the Open American National Corpus, spanning various contexts such as conversations, government reports, speeches, letters, and fiction. To collect sentences, external contributors provide three types: "ENTAILMENT" (must be true), "CONTRADICTION" (must be false), and "NEUTRAL" (no strong connection). This dataset serves as a demanding benchmark for evaluating advanced model performance compared to SNLI. [6].

1.2.3 *QNLI*.

The task of Question Natural Language Inference (QNLI) is to determine whether one group of sentences contains the information required to answer the question posed in the other group of sentences. This dataset is a subset of SQuAD v1.1 and is a question-answering dataset consisting of question-paragraph pairs, where one of the sentences in the paragraph (obtained from Wikipedia) contains the answer to the corresponding question (written by an human)[5].

1.2.4 *QQP*.

The Quora Question Pairs (QQP) dataset consists of over 400,000 question pairs from the community question-answering website Quora. Each question pair is assigned a binary value indicating whether the two questions are paraphrase of each other. The author report that in this dataset, the ground truth values are associated with some degree of noise[3]. The class distribution in this dataset is also unbalanced (63% negative).[5]

1.2.5 *SST-2*.

The Stanford Sentiment Treebank (SST) is the first corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language (how the meaning of the sentence changes the more it is developed). It contains 11,855 single sentences extracted from movie reviews. In the case of the SST-2 dataset, labels are binary and either negative/somewhat negative or positive/somewhat positive. The full corpus has 5 labels: negative, somewhat negative, neutral, somewhat positive or positive. [4]

1.2.6 *CoLA*.

In a 2018 study, artificial neural networks (ANNs) were trained to better understand grammar like human beings. These are trained to make judgments about the grammatical acceptability of sentences. This means distinguishing between well and poorly structured sentences The Corpus of Linguistic Acceptability (CoLA) consists of 10657 sentences from 23 linguistics publications, annotated for acceptability (grammatically) by their original authors. CoLA is the largest corpus of its kind. Grammatic violations included morphological, syntactic and semantic violations

1.2.7 *RTE*.

The RTE (Recognizing Textual Entailment) dataset is a collection of pairs of texts designed for evaluating natural language understanding systems, particularly their ability to determine the logical relationship between two given pieces of text. The task involves deciding whether the meaning of one text (the hypothesis) can be logically inferred or deduced from another text (the premise).

Each pair in the RTE dataset consists of a premise and a hypothesis. The relationship is classified as one of the following: entailment, neutral or contradiction.[5]

2 THE BERT MODEL

BERT-Base, is a state-of-the-art, pre-trained natural language processing (NLP) model developed by Google, with 12 transformers and 110M parameters. Some key features and concepts related to BERT include: (1) Transformer Architecture: BERT is built upon the Transformer architecture, which is a neural network architecture. (2) Bidirectional Training: Unlike earlier models that read text in a unidirectional manner (from left to right or vice versa), BERT uses bidirectional training to understand the meaning of each word in a sentence. This bidirectional training is crucial for capturing context-dependent meanings and relationships between words. (3) Pre-training: BERT is pre-trained on large amounts of text data in an unsupervised manner. During pre-training, it learns to predict missing words in sentences, which helps it capture contextual information and semantic relationships within the text. (4) Transfer Learning: After pre-training, BERT can be fine-tuned on specific downstream tasks with a smaller dataset. This transfer learning approach allows BERT to adapt its learned representations to a more specific NLP tasks, such as text classification, named entity recognition and question answering. (5) Contextual Embeddings: BERT produces contextual embeddings for each word in a sentence, taking into account the surrounding context. This leads to more accurate and nuanced representations of words based on their context in a given sentence.

2.0.1 ***Pre-training.***

(1) Transformer Architecture: BERT utilizes the Transformer architecture, which includes an encoder for processing input text. Unlike directional models, BERT's encoder reads the entire sequence of words at once, making it bidirectional. (2) Training Strategies: BERT uses two training strategies: Masked Language Model (MLM) and Next Sentence Prediction (NSP). (3) Masked Language Model (MLM): Before inputting word sequences into BERT, 15% of the words are replaced with a [MASK] token. The model predicts the original values of these masked words based on the context provided by the non-masked words in the sequence. The BERT loss function considers only the prediction of masked values, enhancing context awareness. (4) Next Sentence Prediction (NSP): BERT receives pairs of sentences as input during training and learns to predict if the second sentence follows the first in the original document. The input is first tokenized. A [CLS] token is added at the beginning of the first sentence, and a [SEP] token is added at the end of each sentence. Sentence embeddings and positional embeddings are used to distinguish between sentences. A sentence embedding indicating belonging to sentence A or B is added to each token. A positional embedding is added to each token to indicate its position in the sequence. (5) The embedded word sequences, including the [MASK] tokens, are processed through BERT's Transformer encoder. Word embeddings are vector representations of words, contained in BERT's that the model can now do mathematical operations. BERT has a Word Embedding Lookup Table consisting of 30,000 tokens and each token has 768 features in its embedding. There is a dictionary that maps the raw word with the hashed word id, that then corresponds to a row in the lookup table. (6) Training Process: BERT trains on both MLM and NSP simultaneously, minimizing the combined loss function of the two strategies.

2.0.2 ***Why Fine-Tune?***

Fine-tuning a model refers to using the weights of an already trained network as the starting values for training a new network. BERT is pre-trained on a corpus consisting of the BooksCorpus (800M words) and English Wikipedia (2,500M words) [1]. For this assignment, the volume of data that BERT is pre-trained on is prohibitively large to replicate. Instead, we have decided to fine-tune the BERT base model on various language datasets, detailed in Section 1, and report the performance.

In general, fine-tuning also offers the advantages of adding words and sentences more specific to a task into the vocabulary, if only a small training dataset is available, fine-tuning BERT on this training set will offering better accuracies and using a pre-trained BERT model can also allow for quicker model development times.

BERT can be specific NLP tasks such as question-answer, named entity recognition, textual entailment, sentiment analysis, paraphrase detection, text summarization, conversational AI, information retrieval, document classification etc.

3 EXPERIMENT DESIGN

We ran BERT-base on the NLP tasks mentioned above with the goal of reproducing the results mentioned in the paper. To fine-tune on GLUE, we use a batch size of 32, and fine-tune for 3 epochs over the validation data for the GLUE tasks. We made use of TPUs for QNLI, SST-2, CoLA, STS-B, MRPC, RTE, MNLI-(m/mm) and QQP datasets. We used a maximum sequence length of 128 to save substantial system memory and a learning rate of 2e-5.

3.0.1 ***Preprocess the text.***

We construct a Keras model for preprocessing text data using the BERT (Bidirectional Encoder Representations from Transformers) model. The function takes a list of string-valued features and an optional parameter for the sequence length of BERT inputs. It creates Keras Input layers for each feature, tokenizes the input text using the BERT preprocessing model from TensorFlow Hub, and optionally trims the tokenized segments to fit the specified sequence length. The function then packs the tokenized segments into a format suitable for input to a BERT model using a Keras layer. The resulting Keras Model, when called with a list or dictionary of string tensors corresponding to the input features, produces a dictionary of tensors ready for consumption by the BERT model. This preprocessing is then applied to all the inputs in the dataset.

3.0.2 ***Define the model.***

We define our model for sentence or sentence pair classification by feeding the preprocessed inputs through the BERT encoder and putting a linear classifier on top (or other arrangement of layers as you prefer), and using dropout for regularization.

3.0.3 ***Train the model.***

To distribute training onto TPU workers, we create and compile our main Keras model within the scope of the TPU distribution strategy.

Preprocessing, on the other hand, runs on the CPU of the worker host, not the TPUs, so the Keras model for preprocessing as well as the training and validation datasets mapped with it are built outside the distribution strategy scope. The call to fit function takes care of distributing the passed-in dataset to the model replicas.

Fine-tuning follows the optimizer set-up from BERT pre-training (as in Classify text with BERT): It uses the AdamW optimizer with a linear decay of a notional initial learning rate, prefixed with a linear warm-up phase over the first 10% of training steps. In line with the BERT paper, the initial learning rate is smaller for fine-tuning (best of 5e-5, 3e-5, 2e-5).

3.0.4 ***Proposed Modifications to the BERT fine-tuning model - Quantization and Pruning.***

Despite its recent success and wide adoption, fine-tuning BERT on a downstream task is prone to overfitting due to overparameterization; BERT-base has 110M parameters and BERT-large has 340M parameters. The overfitting worsens when the target downstream task has only a small number of training examples. [Devlin et al. (2019)] show that datasets with 10,000 or less training examples sometimes fail to fine-tune BERT [1].

To mitigate this critical issue, multiple studies attempt to regularize BERT by pruning parameters or using dropout to decrease its model complexity. In an extension to this report, we will focus on regularizing BERT by pruning attention heads, and then fine-tuning the model again. Quantization involves reducing the precision of the model's weights & activations and pruning involves removing certain connections (weights) from the model, typically those with small magnitudes, resulting in a sparser model. Pruning yields simple and explainable results and it can be used along with other regularization methods.

In order to avoid combinatorial search, whose computational complexity grows exponentially with the number of heads, the existing methods measure the importance of each attention head based on heuristics such as an approximation of sensitivity of BERT to pruning a specific attention head. However, these approaches are based on hand-crafted heuristics that are not directly related to the model performance, and therefore, would result in suboptimal performance. Moreover, all the existing methods cannot find out the optimal number of the attention heads to be pruned. Thus, the number has to be arbitrarily selected even though the optimal number significantly differs depending on the tasks.

4 EXPERIMENTAL DESIGN AND RESULTS

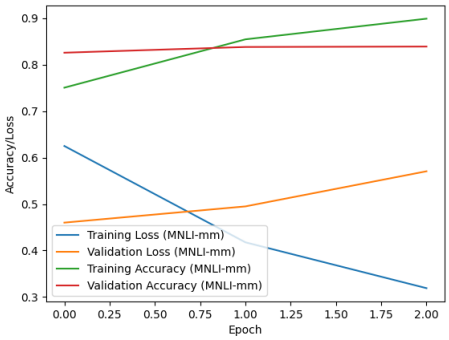

Figure 2: MNLI-mm

The MNLI-mm dataset exhibits high accuracy which correctly corresponds to the values in the original paper. After the first epoch, there is a gentle increase in validation loss, signalling an increase in the loss of the model's confidence.

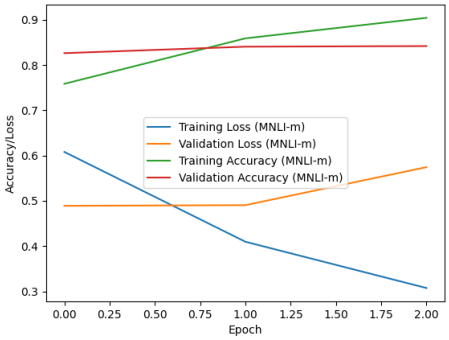

Figure 3: MNLI-m

The MNLI-m dataset exhibits high accuracy which correctly corresponds to the values in the original paper. After the first epoch, there is a gentle increase in validation loss, signalling an increase in the loss of the model's confidence.

The QQP dataset shows fairly consistent F-1 scores as per the original paper although we observe overfitting due to an increase in validation loss. Having a high validation accuracy and a high validation loss indicates that the model is having low confidence it its classification.

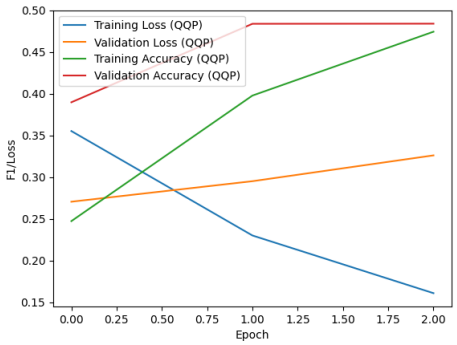

Figure 4: QQP

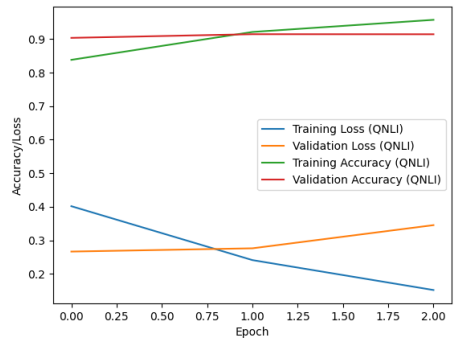

Figure 5: QNLI

The QNLI dataset shows consistent validation accuracy results in accordance with the original paper. Looking closely at the loss curves, we observe that there is overfitting due to the fact that Cross entropy measures how confident you are about a prediction. This eventually means the model isn't confident about the predictions it makes as the epochs progress.

The SST-2 dataset shows extremely accurate results which correctly coincide with the values from the original paper. There is a flatline in validation loss, indicating there is some level of overfitting occurring in the model. Having a high validation accuracy and a moderately high validation loss indicates that there is a decrease in model's confidence.

The CoLA dataset shows considerably higher validation accuracy results that the original paper. There is also a gradual increase in validation loss signaling loss in model's confidence as the epoch progresses.

Metrics	CoLA	SST-2	MRPC	QQP	MNLI(m/mm)	QNLI	RTE
Training Loss	25.6	8.46	28.5	16.7	30.7/31.9	15.2	51.1
Validation Training Loss	64.1	34.6	51.4	31.4	57.4/57.0	35.8	63.6
Accuracy	91.4	97.6	-	-	90.4/89.9	95.6	76.3
Validation Accuracy	82.4	92.7	-	-	84.1/83.9	91.3	63.6
F-1	-	-	43.3	47.8	-	-	-
Validation F-1	-	-	70.8	47.9	-	-	-

Table 1: Performance comparison of BERT models on different tasks after 3 epochs. All experiments make use of sparse categorical cross entropy loss.

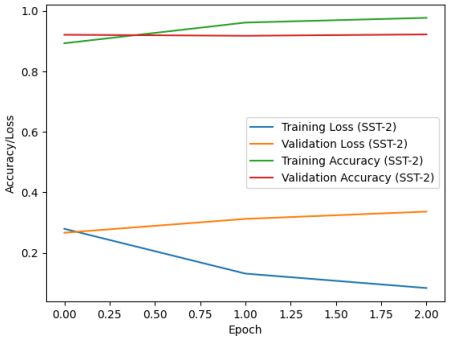

Figure 6: SST-2

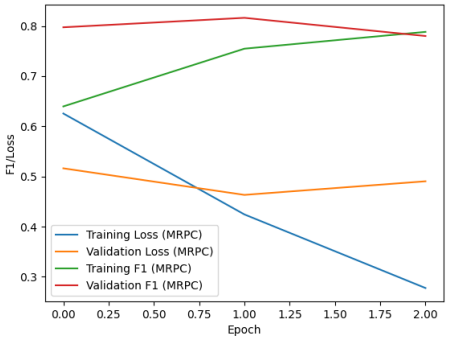

Figure 8: MRPC

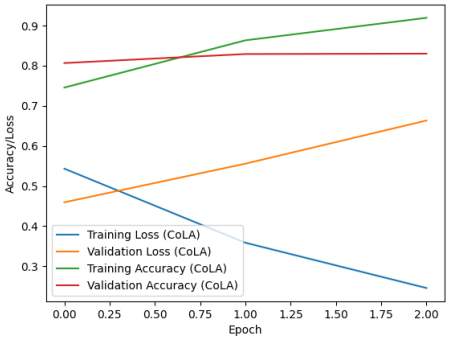

Figure 7: CoLA

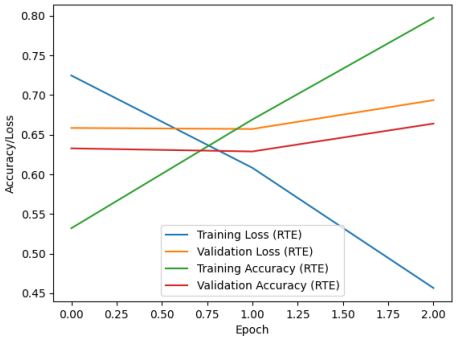

Figure 9: RTE

The MRPC dataset shows a decrease in validation loss but then after the first epoch, it slightly increases. Correspondingly, the validation accuracy decreases after the first epoch as well, signaling misclassifications. A quick way to solve this would be to stop training after the first epoch. Validation F1 scores remain fairly consistent with the original values from the paper.

The RTE dataset shows an accurate validation accuracy as per the original paper. There is a sharp increase in validation loss indicating loss in model's confidence and overfitting.

5 VIABILITY

Figures 14 to 17 are screenshots showing that we were able to fine-tune the model successfully, as well as use the fine-tuned model for some NLU tasks.

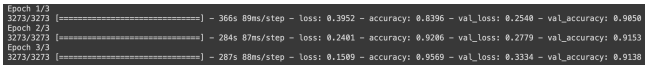

Figure 10: Time taken to Train One Epoch when Fine-Tuned on QNLI Dataset

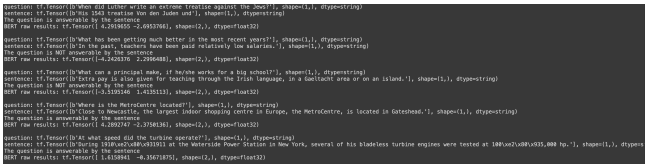

Figure 11: Example of QNLI Dataset Fine-Tuned Model on Question Answering Tasks

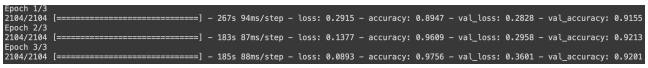

Figure 12: Time taken to Train One Epoch when Fine-Tuned on SST-2 Dataset

Figure 13: Example of SST-2 Dataset Fine-Tuned Model on Sentiment Analysis Tasks

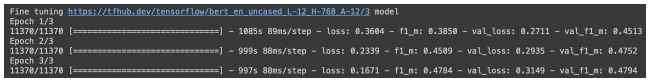

Figure 14: Time taken to Train One Epoch when Fine-Tuned on QQP Dataset

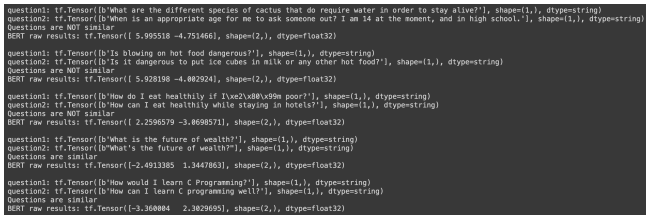

Figure 15: Example of QQP Dataset Fine-Tuned Model on Sentiment Analysis Tasks

Metrics	QNLI	SST-2	QQP
Number of Training Samples	10473	67349	363846
Time Taken to Train One Epoch(s)	336	267	1085
Time Taken to Run the Model on a Test Natural Language Task Set(s)	184	82	136
Test Set Size	5463	1821	390965

Table 2: Metrics for Fine-Tuning BERT on QNLI, SST-2 and QQP Datasets

REFERENCES

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[2] William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*. https://aclanthology.org/I05-5002

[3] Shankar Iyer, Nikhil Dandekar, and Kornél Csernai. [n. d.]. https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

[4] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard (Eds.). Association for Computational Linguistics, Seattle, Washington, USA, 1631–1642. https://aclanthology.org/D13-1170

[5] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. arXiv:1804.07461 [cs.CL]

[6] Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. arXiv:1704.05426 [cs.CL]