BERTille, a multitask BERT model made in France

Stanford CS224N Default Project

Alexis BONNAFONT
Dep. of Energy Science and Engineering, Stanford University
bonnaf@stanford.edu

Malo SOMMERS
ICME, Stanford University
msommers@stanford.edu

Salma ZAINANA
ICME, Stanford University
szainana@stanford.edu

Abstract

In this project, we aim to replicate the core components of the BERT model, focusing on the multi-head self-attention mechanism and transformer layers, to perform sentiment analysis on the Stanford Sentiment Treebank and an additional dataset of movie reviews. Building on this, we extend our model to encompass multiple downstream tasks, including paraphrase detection and semantic textual similarity, with the goal of creating versatile and generalizable sentence embeddings. Recognizing the potential of multi-task learning, we leverage shared parameters to reduce the number of required parameters and address the challenges of optimizing multiple tasks concurrently. Inspired by Stickland and Murray (2019), we implement projected attention layers (PALs) and a novel training scheduling techniques. Our multi-task model is trained and evaluated on the Stanford Sentiment Treebank (SST), the Quora Dataset, and the SemEval Benchmark Dataset, achieving a baseline score of 0.672 on the dev sets. Through the evaluation of various fine-tuning approaches and their interactions, we improve our model's performance, attaining a development score of 0.776 and a test score of 0.775. Furthermore, we explore a combination of PAL scheduling and Gradient Vaccine, which significantly accelerates training during the initial epochs of fine-tuning.

1 Key Information to include

- **Mentor:** Josh Singh
- **Contributions:** Given Appendix B

2 Introduction

Natural Language Processing (NLP) has made significant strides, achieving impressive results in tasks like sentiment analysis, paraphrase detection, and semantic similarity. While models are often trained for a single task, multitask learning can be advantageous for applications like multilingual translation. This study explores adapting a single, large base model to handle multiple tasks effectively. This paper tackles the challenges of multitask classification with BERT for sentiment analysis, paraphrase detection, and semantic similarity scoring. The main difficulties include the complexity of natural language and the need to capture subtle semantic nuances. Additionally, the extensive trainable parameters in models make adapting to new tasks computationally demanding.

Recent advancements in natural language processing have utilized models like BERT to tackle various tasks effectively. For example, Wang et al. (2021) shows that BERT-SAN significantly improves aspect-based sentiment analysis. Similarly, Ko and Choi (2020) introduces Paraphrase-BERT, which uses multitask learning to enhance paraphrase identification by sequentially learning question answering and paraphrase tasks. Additionally, Li et al. (2022) presents LP-BERT, designed for semantic network completion in knowledge graphs, achieving state-of-the-art results through multitask pre-training and innovative data augmentation. These studies demon-

strate the potential of multitask learning and advanced pre-training methods to enhance model performance in various NLP tasks.

In this work, we propose an effective approach to adapt BERT for multitask learning, focusing on two main steps. First, we handle multitasking by updating BERT for different tasks through task scheduling, gradient treatment methods, and regularized optimization. Second, we optimize our multitask classifier's architecture to enhance performance without significantly increasing parameters by efficiently managing dual inputs and integrating projected attention layers. Our contributions in task scheduling and architectural tuning demonstrate significant improvements in multitask learning with BERT, providing a robust framework for various NLP tasks.

3 Related Work

Our approach builds on several key advancements in the field of multitask learning and optimization techniques.

The foundation of our project lies in the original BERT model, which has been instrumental in advancing natural language processing tasks. BERT utilizes a multi-head self-attention mechanism and transformer layers to achieve state-of-the-art performance across various tasks.

Stickland and Murray (2019) proposed a method for multitask learning using BERT, specifically addressing the challenge of imbalanced datasets. Their annealed sampling method dynamically adjusts the frequency of task sampling based on the number of training examples, ensuring a balanced contribution to the training process. This technique inspired our adaptive scheduling strategy to mitigate dataset imbalance.

Competing gradients are a known issue in multitask learning, where gradients from different tasks can conflict during optimization, leading to suboptimal performance. Techniques like gradient normalization and gradient surgery help address these conflicts. The gradient surgery technique, as described by Wang et al. (2020), projects conflicting gradients onto a common plane to align them, reducing conflicts and enhancing performance. We also explore the gradient vaccine extension, which dynamically reweights gradients based on their contribution to the overall loss.

Stickland and Murray (2019) also introduced the Projected Attention Layer (PAL), a low-dimensional multi-head attention layer added in parallel to normal BERT layers which is specific to each task. With PAL we introduce task-specific global attention layers, tailored to each task, enhancing BERT's performance without significantly increasing the number of parameters as most of the are shared in the classical Bert multi-head attention layers.

4 Approach

Our project aims to replicate the BERT model and extend it for multitask learning. The primary tasks include implementing the multi-head self-attention mechanism and transformer layers, fine-tuning BERT for sentiment analysis, and enhancing it for multiple downstream tasks.The second step focuses on adjusting the architecture to improve performance without significantly increasing the number of parameters.

4.1 BERT model: baseline version

The BERT model we implemented consists of 12 layers, each comprising a series of operations including normalization, self-attention, and feed-forward processing. Each layer begins with normalization and a self-attention mechanism, which allows the model to focus on different parts of the input. The self-attention mechanism includes another normalization step followed by multi-head attention, enabling the model to simultaneously examine various aspects of the input. The output of the self-attention process is then passed through a feed-forward network to produce the final output for the layer.

Our baseline approach involves fine-tuning a multitask BERT model to achieve optimal performance across various tasks. We leverage the original BERT implementation with pre-trained weights and a standard sentiment classification pipeline. Initially, a simple classifier was implemented, comprising a BERT model with a dropout and a linear layer on top, trained for each task. Fine-tuning is performed using pre-trained weights on the SST and CFIMDB datasets, following hyperparameters inspired by the original minBERT framework. Sentiment analysis and paraphrase detection are approached as classification problems, while semantic similarity is addressed through regression. The model undergoes pre-training where only the classification layers are updated, followed by fine-tuning all parameters using a random task scheduler that selects the training task at each step.

4.2 Multi-Tasking Optimization

Independent Task-Specific Pretraining Inspired by related work Wei et al. (2021), we implemented independent task-specific pretraining.

During this phase, BERT parameters are kept frozen to ensure classifiers for each task remain independent. This approach involves pretraining the multitask classifier on each task individually with the frozen BERT layers. By training the classifier separately for each task, we could select the top-performing parameter sets for each fully connected layer. This technique enhances overall performance by allowing specialized learning without task interference.

The scheduling task In our multi-task BERT model, we encountered a significant imbalance in the number of training examples between paraphrase detection and sentiment classification tasks. Specifically, we had more training examples for paraphrase detection compared to those for sentiment classification. This imbalance posed a challenge, as the model tended to prioritize the paraphrase detection task during training, potentially leading to suboptimal performance on the sentiment classification task.

To address this issue, we adopted the task scheduling approach proposed by Stickland and Murray (2019) Stickland and Murray (2019), particularly the annealed sampling method described in section 4.1 of their paper. This method involves dynamically adjusting the frequency of task sampling based on the number of training examples for each task, ensuring a more balanced contribution to the training process.

We implemented an adaptive scheduling strategy to address dataset imbalance, ensuring the model allocated adequate learning capacity to both tasks. This balanced training regimen gave sufficient attention to the smaller sentiment classification dataset, resulting in improved overall performance and demonstrating the effectiveness of dynamic sampling in multi-task learning.

We also tested the Round Robin scheduler which had not as good results as the this one.

Competing gradient In this work, We addressed the issue of competing gradients, which occurs when gradients from different tasks conflict during optimization. We adopt the gradient surgery technique following Wang et al. (2020). Gradient surgery involves projecting conflicting gradients onto a common plane to align them. The update rule is:

$$g_i' = g_i - \frac{g_i \cdot g_j}{\|g_j\|^2} g_j$$

where g_i and g_j are the gradients from different tasks.

We also explore the gradient vaccine extension. Unlike gradient surgery, gradient vaccine mitigates conflicts by dynamically reweighting the gradients based on their contribution to the overall loss. This approach can be more effective for multitask BERT classification as it better preserves task-specific learning while resolving conflicts. The update rule for gradient vaccine is:

$$\mathbf{g}_i' = \mathbf{g}_i + \frac{\|\mathbf{g}_i\|(\phi_{ij}^T\sqrt{1-\phi_{ij}^2} - \phi_{ij}\sqrt{1-(\phi_{ij}^T)^2})}{\|\mathbf{g}_j\|\sqrt{1-(\phi_{ij}^T)^2}} \cdot \mathbf{g}_j. \tag{1}$$

Where ϕ_{ij} is the cosine similarity between gradients g_i and g_j. For this, we used the code written by Antoine Nzeyimana accessible via GitHub [1].

Combination of scheduling method and gradient vaccine: Grad-Scheduling Gradient Surgery requires calculating one gradient per task, which means that tasks must be processed in a specific order, making it difficult to use scheduling methods which aim to allocate learning capacity based on task importance or dataset size, but Gradient vaccine's requirement for sequential task processing conflicts with this approach, limiting the ability to adaptively balance training across multiple tasks.

To address the issue of incompatibility, the model is updated after processing several batches and selects the task for each batch using a modified schedule. The process begins with a predefined list of tasks. At each epoch, a value is calculated to adjust the focus on each task based on the dataset size and the current training stage. Probabilities are computed and normalized to determine how often each task should be chosen. These probabilities are then used to randomly select tasks for each batch, ensuring a balanced and fair training regimen. The selected tasks are shuffled, and the losses for each task are updated accordingly.

This approach maintains a balanced learning process across different tasks, adjusting dynamically to the dataset sizes and training progress, ultimately improving overall performance.

4.3 Model design tuning

Projected Attention Layers (PAL) In our approach, we modified the BERT layers by introducing a new attention term tailored to each task, which only requires a few additional parameters. This approach allows us to retain most of the shared attention parameters within the self-attention mechanism applied to the embedding

[1] https://github.com/anzeyimana/Pytorch-PCGrad-GradVac-AMP-GradAccum

of the input token h $SA(h)$ while fine-tuning the attention mechanism for each task through the addition of $PAL(h)$ in the BERT layer (BL) given by :

$$BL(h) = LN(h + SA(h) + PAL(h))$$

with the task-specific attention expressed as $PAL(h) = V^D g(V^E h)$.

In this study, we opted for a simpler version of the Projected Attention Layer as described in the appendix A, figure 4. We implemented a low-rank multi-head attention layer with task-specific V^D and V^E matrices, ensuring that attention is not shared across layers.

Multiple inputs handling Choosing how to manage inputs is a key in model decision. In this project, we are facing the issue of handling two input sentences required by both paraphrase detection and semantic textual similarity to generate a single output. There are two primary methods: (1) merging both sentences with a separation token and inputting the combined sequence into BERT, or (2) running each sentence separately through BERT, followed by a concatenation, or cosine similarity technique. The Experiment section details the results of testing these methods.

4.4 Memory optimization

Given the large size of our model, our GPU struggled with small batch sizes, causing slow and irregular training updates. To address this, we implemented two optimizations: Automatic Mixed Precision to reduce memory usage, and Gradient Accumulations to simulate larger batch sizes and speed up training.

5 Experiments

5.1 Data

We used the datasets provided: **Stanford Sentiment Treebank (SST)** and **CFIMDB** for the minBERT implementation (single task on sentiment analysis implementation). And **Stanford Sentiment Treebank (SST)** for sentiment analysis, **Quora Question Pairs** for paraphrase detection and **STS Benchmark** for similarity scores for the multitask implementation.

5.2 Evaluation method

We used the same metrics as the ones used to by the leaderboards: number of correctly classified data over the total for paraphrase detection and sentiment analysis and Pearson correlation of the true similarity values against the predicted similarity values for semantic textual similarity. The total score is $(\frac{\text{sentiment accuracy}+\text{paraphrase accuracy}+(\text{sts correlation}+1)/2}{3})$.

5.3 Experimental details

As model configuration, we used, for sentence embedding, the output of the Bert model for sentiment analysis. For paraphrase detection and similarity scores, merging both sentences with a separation token and inputting the combined sequence into BERT was preferred as the results were far better than the ones obtained by running each sentence separately through BERT, followed by a concatenation, or cosine similarity technique. For each task, we have then a linear hidden layer of size 768 (same dimension as our BERT embedding) with dropout (rate 0.2) and a ReLU activation function before a final linear layer, with dropout again, to get the output size wanted: 5 for semantic similarity and 1 for the rest. When used, the projected attention layers (PAL) consisted in 12 attention heads of dimension 11 each.

Our Training phase consisted in 3 sub-phases, that we will breakdown in order:

- Individual pretraining: We first had an individual pretraining phase (described in section 4.2), of 3 epochs with a learning rate of 10^{-3}. We tried to increase the number of epochs with a patience of 3 but it only affected the accuracy of paraphrase detection (stopped after 20 epochs), as the other tasks we early stopped before 3 epochs. Moreover, having 20 epochs didn't changed the results of the fine-tuning phase which made us chose 3 epochs. The batch size varied depending on the methods and the task, allowing for different batch sizes for each task with gradient accumulations. Generally, the batch size was set to 16 for paraphrase detection, and 32 for SST and STS. This phase took 15 minutes.

- First fine-tuning phase: We then had a first fine-tuning phase of 10 epochs with a learning rate of 10^{-5} and a patience of 3, using pal scheduler without gradient surgery. This phase was without having projected attention layers as adding them at the beginning of fine-tuning gave us low results. The batch size was the same as previously. This phase took 1 hour (6 minutes per epoch). During this phase we also tired to use gradient compromise combining pal scheduler with gradient vaccine.

- Second fine-tuning phase: Finally, we had a last fine-tuning phase of 10 epochs again with a learning rate of 5×10^{-6}, a patience of 3, using gradient vaccine (round-robin scheduler) and that we tested with and without adding projected attention layers. For memory limits (16GB GPU),

the batch size for this phase was 8 for al task. This phase took 3 hours (18 minutes per epoch). During this phase we also tired to use gradient compromise combining pal scheduler with gradient vaccine.

5.4 Results

Our dev accuracies for different methods are given table 1.

On the test set, our best model provided the following accuracy table 2.

Firstly, we observed that concatenating sentences before embedding, rather than after, significantly increased our scores and made the addition of a linear layer more effective. The independent pretraining phase also improved accuracy, particularly by reducing overfitting on the STS and SST datasets when pretraining tasks independently instead of jointly with the Quora dataset. There is a noticeable difference between adding PAL layers from the beginning or after an initial fine-tuning phase. However, PAL layers did not improve our results as expected. We conducted an analysis (too lengthy to include in the report) to determine if the predictions were different (i.e., if the model was able to detect different aspects), but it appears that the PAL model performed worse (both models made errors on the same entries, but the PAL model was "more wrong"). Additionally, during the second fine-tuning phase with PAL, patience was reached at epoch 6 (improvement ceased after 3 epochs), which may have led to lower accuracies. Finally, despite significantly increasing training speed, Grad-Scheduling did not outperform PAL scheduling and gradient vaccine, the two methods it was supposed to merge.

6 Analysis

6.1 Paraphrase Detection

In this report, we present several visualizations to evaluate the performance of our model at different stages of training. In Appendix A, Figure 5 shows the confusion matrices that depict the model's performance in paraphrase classification and sentiment analysis tasks. For paraphrase classification, these matrices show the distribution of predicted classes compared to the true classes after different training stages. We see that for this task, the model is already efficient after the 1st fine-tuning.

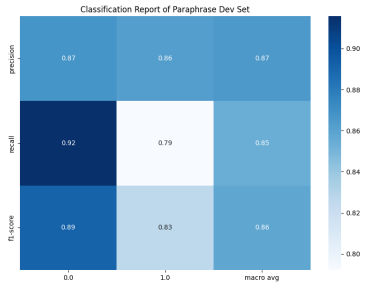


Figure 1: Classification Report for Paraphrase Analysis

The classification report for paraphrase analysis provides detailed metrics such as precision, recall, and F1-score for each class, giving insights into the model's performance across different sentiment classes. Our seems to perform better on the first class (not paraphrase) with a f1 score of 0.89, and is a bit less proficient to find all the paraphrase, with a recall for the second class of 0.79.

Moreover here's the analysis of some predicted outputs:

Good Prediction: "What is the peace sign in Australia, and how did it come to be the peace symbol?" and "Write something about your team?" - Correctly identified as not paraphrases.
Good Prediction: "Is it biologically good or bad to marry other caste?" and "Is it good or bad to marry other caste?" - Correctly identified as paraphrases.
Bad Prediction: "Is it good to go for an MBA in construction management from RICS, Amity University?" and "What would be the approximate amount to pursue MBA in construction management from RICS Amity including hostel facilities as well?" - Incorrectly predicted as paraphrases.
Bad Prediction: "I want to buy a new laptop with i5 processor, 8gb RAM, 2gb graphic card, 1tb hard disk. Which one should I buy?" and "How much does it cost to buy a laptop with this configuration: i5, 8gb RAM, 1tb hard disk, 2 GB graphic card?" - Incorrectly predicted as not paraphrases.

The model succeeded in identifying paraphrases when sentences were either distinctly different or nearly identical in wording. However, it failed when sentences shared similar context but diverged significantly in their focus or specific details.

Method	Dev. SST	Dev. Paraphase	Dev. STS	Dev. acc
Random	0.199	0.525	0.015	0.408
BERT + concat emb.	0.480	0.699	0.368	0.621
BERT + concat emb. + cosine similarity	0.458	0.708	0.699	0.672
Basis = BERT + concat sentences	0.462	0.740	0.743	0.691
Basis + Pal schedule + 1 hidden	0.504	0.873	0.856	0.768
Basis + Pal schedule + 1 hidden + indiv. pretrain	**0.527**	**0.866**	**0.872**	**0.776**
Basis + Pal schedule + 1 hidden + indiv. pretrain + PAL	0.245	0.620	0.526	0.543
Basis + Pal schedule + 1 hidden + indiv. pretrain + PAL only during 2nd fine-tuning	0.504	0.854	0.873	0.765
Basis + 1 hidden + indiv. pretrain + Grad-Scheduling	0.513	0.832	0.845	0.756

Table 1: Dev accuracies for different techniques

Method	Dev. SST	Dev. Paraphase	Dev. STS	Dev. acc
BaseModel + Pal schedule + 1 hidden + indiv. pretrain	**0.520**	**0.867**	**0.878**	**0.775**

Table 2: Test accuracies for our best model

6.2 Sentiment Analysis

Similar to the paraphrase classification, the confusion matrices for sentiment analysis (appendix A, figure 6) illustrate the accuracy of the model in predicting the sentiment classes at different training stages. Those matrices are quite interesting as they highlight the initial model's tendency to predict neutral sentiments. After pretraining, the model predominantly makes "average" predictions, which evolve significantly during the fine-tuning phase. This evolution underscores the effectiveness of the second phase of fine-tuning, leading to more accurate sentiment predictions.

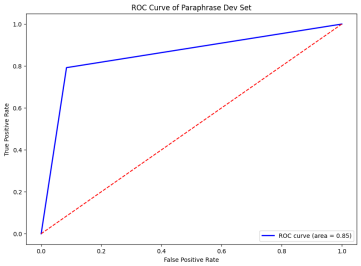


Figure 2: ROC Curve for Paraphrase Analysis

The ROC curve for paraphrase analysis illustrates the trade-off between true positive rate and false positive rate, providing insights into the performance of the model in binary classification tasks. We have a 0.85 ROC score, which indicates an overall good performance of our model.

Moreover here's the analysis of some predicted outputs:

Good Prediction: "A warm, funny, engaging film." - Predicted 4, truth 4.

Moderate Prediction: "Uses sharp humor and insight into human nature to examine class conflict, adolescent yearning, the roots of friendship, and sexual identity." - Predicted 3, truth 4.

Bad Prediction: "A coda in every sense, The Pinochet Case splits time between a minute-by-minute account of the British court's extradition chess game and the regime's talking-head survivors." - Predicted 1, truth 4.

Bad Prediction: "It takes a certain kind of horror movie to qualify as 'worse than expected,' but Ghost Ship somehow manages to do exactly that." - Predicted 3, truth 0.

The model performed well on clear, straightforward sentiments. It struggled with complex or nuanced reviews, often failing to fully capture the underlying emotional tone, either underestimating positive sentiment or misinterpreting strong negative sentiment.

6.3 Similarity Analysis

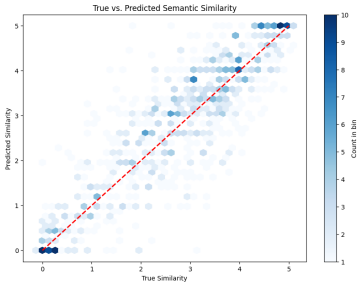


Figure 3: Semantic Similarity Binplot

The semantic similarity binplot displays the distribution of semantic similarity scores, providing an overview of the model's performance in measuring similarity between different phrases. We see that our prediction are around the identity line (indicating perfect predictions), and a bit above, showing us that our model tends to over-predict (predicting in average higher similarity than what expected).

Moreover, here's the analysis of some predicted outputs:

Good Prediction: "The boy is playing the piano." and "The woman is pouring oil into the pan." - Predicted 0.03361, truth 0.
Good Prediction: "Two people are playing golf on a golf course." and "Two people are on a golf course playing golf." - Predicted 4.979195, truth 5.
Bad Prediction: "Yes, you should mention your experience." and "Yes, you should make a résumé." - Predicted 0.023297, truth 2.
Bad Prediction: "It's also a matter of taste." and "It's definitely just a matter of preference." - Predicted 1.873347, truth 5.

The model was accurate when sentences were clearly unrelated or nearly identical. It encountered difficulties with sentences that were contextually similar but had different specific meanings or intentions, leading to inaccurate similarity scores. Moreover in the "resumé" example it couldn't capture the meaning of this word in this context.

7 Conclusion

In this project, we built upon existing advancements to propose an enhanced and effective BERT model for multitask learning. We explored various optimization techniques, including independent training, PAL scheduler, gradient vaccine, and their combinations. Additionally, we investigated architectural design enhancements, such as adding PAL layers to our model. Throughout our work, we gained insights into the importance of proper data handling and the impact of imbalanced data on the effectiveness of the methods we explored.

In future work, we plan to investigate the use of data augmentation techniques tailored to each task. This approach aims to enhance model generalization and performance, particularly for tasks with limited training data. Additionally, we intend to explore transfer learning techniques to leverage knowledge acquired from related tasks or domains.

8 Ethics Statement

Biases in sentiment analysis can perpetuate stereotypes or distort individuals' opinions, leading to ethical concerns about the fairness and accuracy of algorithmic decision-making. Bias can arise from training data, which may contain racial, gender, or other stereotypes, as well as data selection, where certain populations or opinions may be overrepresented or excluded. Human annotators can also introduce bias when creating datasets, which impacts model performance.

Another societal risk relates to the reliance on AI models such as BERT for sentiment analysis, which could influence how sentiments are interpreted and communicated, potentially impacting the social perceptions of different important matters such as new public policies, new ideologies, ect. AI models may amplify extreme or minority opinions if those opinions are overrepresented in the training data. This amplification can affect public perception and lead to increased polarization of opinions.

To mitigate these risks, we would implement bias detection and mitigation techniques during model training and evaluation to address fairness and accuracy concerns. These techniques include the use of data rebalancing methods, regularization to minimize bias, and regular model audits to identify and correct emerging biases. Additionally, we would transparently report any limitations or biases observed in our model results so that users are aware of the potentials and limitations of our sentiment analysis system.

References

Bowon Ko and Ho-Jin Choi. 2020. Paraphrase bidirectional transformer with multi-task learning. In *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 217–220. IEEE.

Da Li, Sen Yang, Kele Xu, Ming Yi, Yukai He, and Huaimin Wang. 2022. Multi-task pre-training language model for semantic network completion.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning.

Yuqi Wang, Qi Chen, and Wei Wang. 2021. Multi-task bert for aspect-based sentiment analysis. In *2021 IEEE international conference on smart computing (SMARTCOMP)*, pages 383–385. IEEE.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2020. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models.

Tianwen Wei, Jianwei Qi, and Shenghuan He. 2021. A flexible multi-task model for bert serving. *arXiv preprint arXiv:2107.05377*.

A Appendix

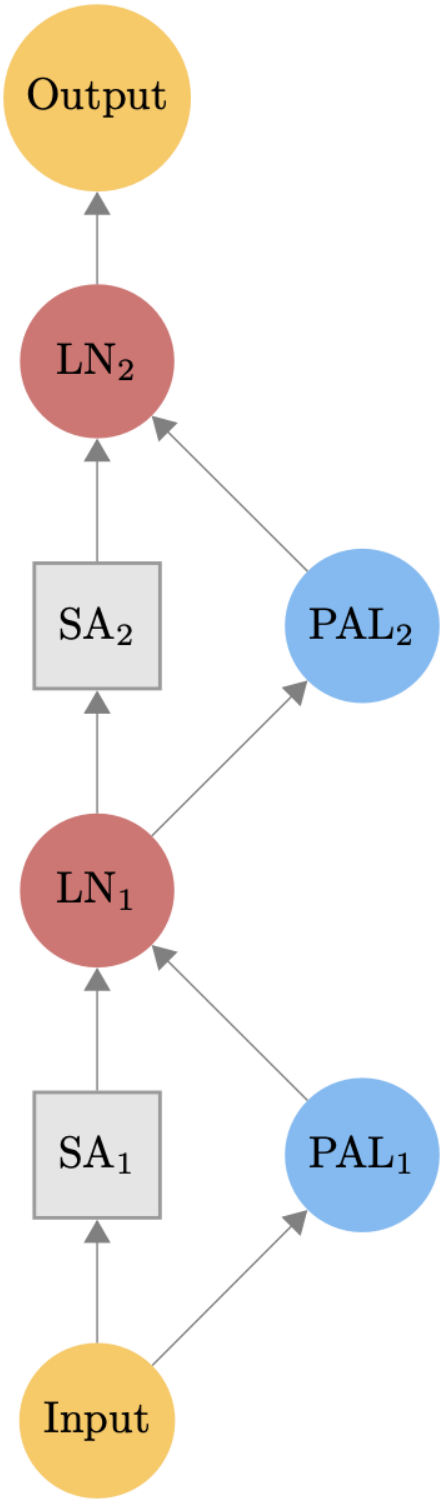
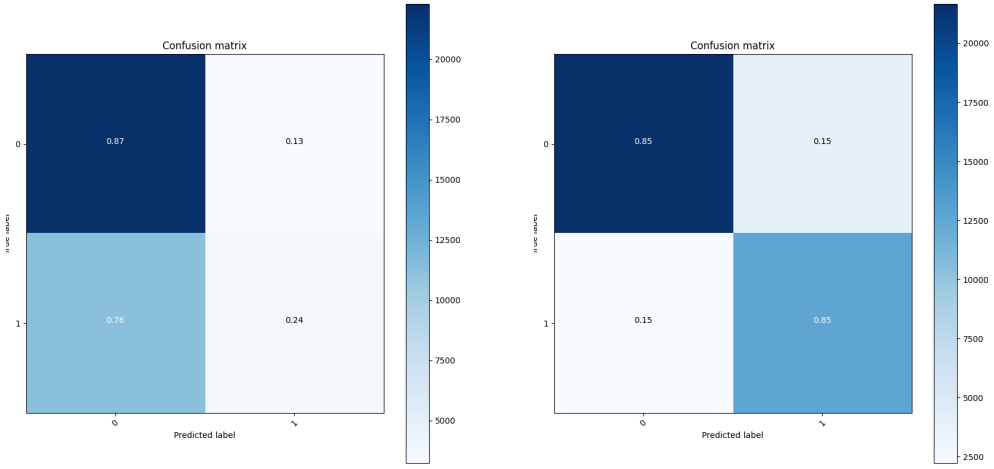
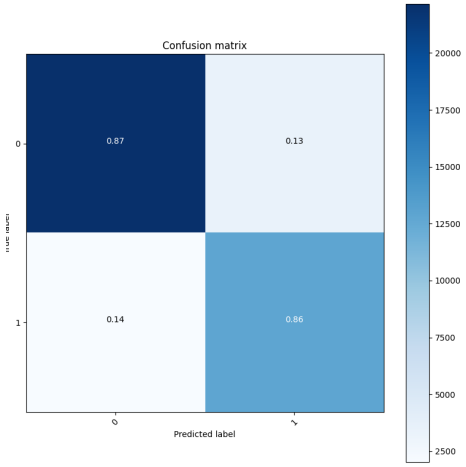


Figure 4: The 'Projected Attention Layer' (PAL) from Stickland and Murray (2019)



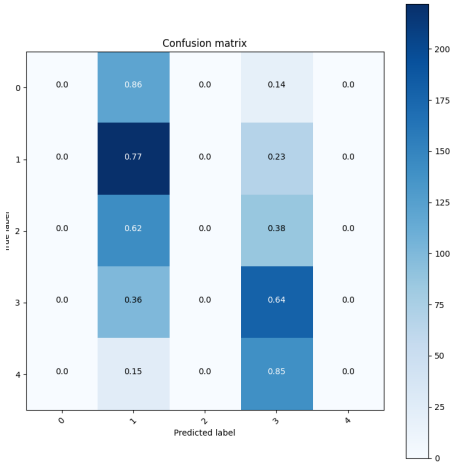
(a) Confusion Matrix after Pretraining for Paraphrase

(b) Confusion Matrix after 1st Finetuning for Paraphrase

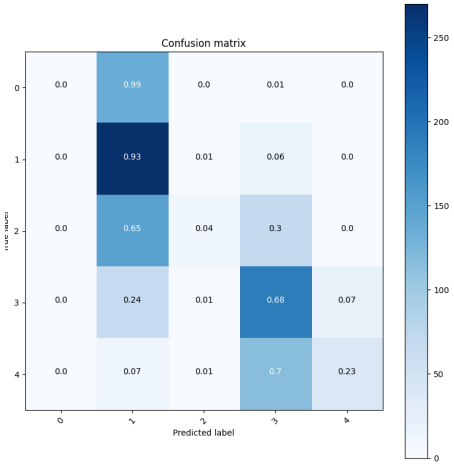


(c) Confusion Matrix after 2nd Finetuning for Paraphrase

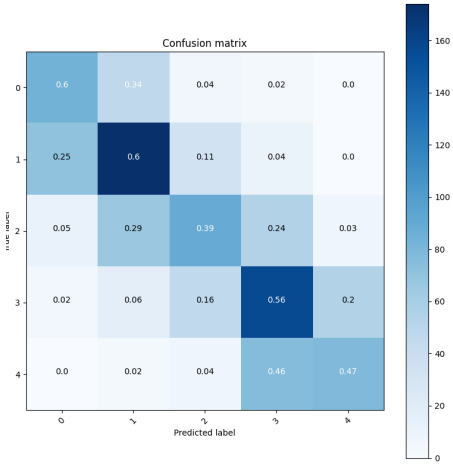
Figure 5: Paraphrase Classification Confusion Matrices



(a) Confusion Matrix after Pretraining for Sentiment Analysis



(b) Confusion Matrix after 1st Finetuning for Sentiment Analysis



(c) Confusion Matrix after 2nd Finetuning for Sentiment Analysis

Figure 6: Sentiment Analysis Confusion Matrices

B Contributions

The responsibilities were distributed as follows:

Malo:

- Code implementation, including individual pretraining and fine-tuning stages.
- Exploration of related articles to provide a strong theoretical foundation.
- Contributing to the minBERT implementation.

Salma:

- Drafting and editing the poster and report.
- Analyzing and interpreting the results.

- Collaborating on the minBERT implementation.

Salma:

- Assisting with the implementation of the PAL algorithm.
- Development of gradient descent and scheduling algorithms.
- Conducting results analysis.