Author: HAMID Baizid Al

Student ID: 2600220455-9

Institution: Ritsumeikan University

Date: 18 January, 2024

Analyzing Full-Frozen, Half-Frozen, and Non-Frozen Layer Approaches in Fine-Tuning BERT Models for Scientific Abstract Classification

1 Introduction

In the realm of Artificial Intelligence and Natural Language Processing, the advent of BERT marked a revolutionary progress. Developed by Google AI, BERT's architecture leverages the Transformer model, distinguishing itself through its bidirectional training approach. Unlike previous models predominantly trained to read text in a singular direction, BERT's bidirectional nature allows it to gather context from both preceding and succeeding text, offering a more nuanced understanding. This breakthrough has significantly advanced the performance benchmarks in a wide array of NLP tasks, particularly in text classification. Its proficiency in capturing complex language nuances makes BERT an ideal choice for classifying scientific abstracts, where precision and contextual understanding are the top priority. Keeping that in mind, this project is presented with a dataset that consists of 1185 different abstracts across three separate categories. This project aims to perform a text classification on the dataset to predict which category a particular abstract belongs to. Furthermore, this project would utilize the fine-tuning of BERT model three separate times to compare their performance through four metrics: accuracy, precision, recall, and F1 score. These metrics will be used to determine which fine-tuning method produces the best result at the end. In Table 1, a general outlook of each layer in the BERT-Base-Uncased model's transformer layer is described in brief.

Table 1: BERT-Base-Uncased model's transformer layers

Transformer Layer	Function
1	Begins understanding word meanings and positional context.
2	Understands basic grammar and forms simple word associations
3	Builds upon grammar and word associations, grasping more complex sentence structures.
4	Deepens understanding of sentence structure and complex word relationships.
5	Develops grasp of language syntax and nuances of words and phrases.
6	Captures abstract language features, including initial semantic understanding.
7	Enhances semantic understanding and contextual meanings.
8	Refines context understanding and sentence relationships.
9	Develops deeper understanding of language nuances and idiomatic expressions.
10	Further refines complex language structure handling and ambiguity management.
11	Synthesizes lower-level understanding into coherent wholes, handling complex relationships.
12	Integrates all features together and outputs final embeddings.

2 Methodology

Python is used as the primary programming language for this project as it widely supports machine learning and deep learning development. The libraries used for this project include PyTorch, Numpy, Hugging Face Transformers, Pandas, Scikit-learn, TQDM and Matplotlib. These libraries will be explained thoroughly in the next sections. For achieving experimentation and analysis, the BERT model was chosen exclusively for fine-tuning.

2.1 Data Collection

In this study, the dataset comprised scientific abstracts collected from the arXiv repository, an open-access archive containing a vast array of academic papers across multiple disciplines. The abstract collection process was automated using a Python script, ensuring an efficient and comprehensive approach to data acquisition. The script's design targeted three specific academic fields: Physics, Medicine, and Cyber-Security. A total of 395 unique papers were targeted across each fields, totalling to 1185 abstracts. The data collection was utilized using the 'arXiv' Python library to interface with the arXiv API. For each field, a distinct search query was constructed. Finally, to ensure a comprehensive dataset, a diverse array of subcategories and keywords were included, covering various disciplines of each fields. For achieving this, the python script fetched up to 100 recent papers per query. For each paper, the script extracted critical information like title, authors, abstract, and the URL of the paper, and the URL identifier was used to avoid duplicates in the dataset. Finally the information was then written into three separate CSV files which was designated for each field. In this data collection process, a filter was incorporated that included only those abstracts with a minimum word count of 50, ensuring a substantive level of detail for classification purposes.

2.2 Data Preprocessing

Once the abstracts were collected, a separate preprocessing python script was employed to clean and tokenize the abstracts. This was achieved by first concatenating the datasets from the three fields into a single DataFrame after manually getting rid of all the columns from the collected CSV files, except for the columns of the category and the abstracts themselves. Secondly, text cleaning was performed to remove special characters and numbers which are common in scientific papers but irrelevant for NLP tasks. And finally, the text was then converted to lowercase to maintain uniformity. Furthermore, using NLTK which is a popular NLP library, the preprocessing script further applied lemmatization to the text. This process involved reducing words to their base or dictionary form, which is crucial for minimizing data sparsity and enhancing model performance. The employed script also removed the stopwords or the common words that are typically irrelevant to the analysis to focus only on more meaningful words in the abstracts. Also, as scientific abstracts often contain LaTeX expressions, a unique approach is adopted to handle these expressions as well. It involved identifying LaTeX expressions using a RegEx pattern and temporarily replacing them with a placeholder and, after standard text cleaning was performed on the text, the placeholders were replaced back with the original LaTeX expressions. Thus, this crucial step preserved the mathematical and scientific integrity of the abstracts while also allowing for standard text processing at the same time as well. The final step in preprocessing was tokenizing the cleaned text using the BERT tokenizer. In this preprocessing phase, the tokenizing step converted the raw text into a format that's more suitable for input into the BERT model. This tokenization involved breaking down the text into words, subwords and symbols called tokens. So, after the tokenized text was successfully written onto a new CSV file in which each entry represented an abstract that has been converted into a sequence of token IDs, the file would go on to serve as the main input for the all the BERT model training phases for the subsequent stages of the project.

3 Experiments

As the focus of this paper is to undertake a comprehensive analysis of three fine-tuning strategies applied to BERT in the context of abstract classification, in this section, we delve into the experimental procedures undertaken for training and fine-tuning three variants of the BERT model.

3.1 Fine-Tuning

The fine-tuning for any of the Non-Frozen, Full-Frozen or Half-Frozen of the BERT methods started with loading in the previously saved CSV file of the tokenized dataset of the scientific abstracts. This process included using the BERT tokenizer again, but this time serving a slightly different purpose. Here, in the fine-tuning and training phase, the BERT tokenizer was employed to prepare the tokenized text for input into the BERT model. This process involved of truncating or padding the sequences to a fixed length which in this case was 512 tokens. This step is really important and necessary because BERT requires all inputs to be of the same length. After that, the attention masks were created, and these basically told the model which parts of the input were actual tokens and which were padding. Finally, the labels of the abstracts were converted into a format suitable for model training which in this case was - Physics: 0, Medicine: 1 and, Cybersecurity: 2 respectively.

3.1.1 Learning-Rate and Optimizer

The choice of AdamW as an optimizer for all the mentioned BERT fine-tuning methods was crucial. As AdamW works as an optimizer with a 'weight decay fix', it was used in creating faster convergence and better generalization. The learning rate which in every case was 5×10^{-5} controlled the step size during gradient descent, balancing between too small steps which would lead to a long training time, and too large steps which would cause overshooting of minima. A carefully chosen learning rate also ensured that the model converges to a suitable minimum in the loss landscape, enhancing its classification accuracy. Here for every case, an epsilon value of 1×10^{-8} were also up set to optimize the model's parameters effectively by preventing division by zero during the optimizer updates. Now, the learning rate scheduler allowed for fine-tuning of the model with greater precision as the training progressed, by often decreasing the learning rate to allow finer adjustments to the weights. This approach helps in stabilizing the training process as it neared convergence. In every case of the fine-tuning methods, a linear scheduler was used which basically worked by gradually decreasing the learning rate over time, allowing for more precise adjustments to the weights and avoiding oscillations that can occur with a constant or too aggressive learning rate. The use of a linear scheduler aligned with the goal of three mentioned methods of fine-tuning as well, where subtle adjustments are often required as the model starts fitting closely to the training data.

3.1.2 K-Fold Cross Validation and Training-Validation Loop

After specifying the number of samples in each batch for both training and validation as 32, the parameters for the K-Fold Cross-Validation was set up. For each of the mentioned training and fine-tuning methods, the overall process involved the utilization of 4 epochs each over a 5 fold Stratified K-Fold Cross-Validation which resulted in a 80:20 Training and Validation split. The loss function for multi-class classification for training was defined during this stage as well. The K-Fold Cross-Validation itself was set up by initializing a loop over each fold, which created training and validation subsets for the current fold and initiated the training and validation loops for each epoch as well. After the optimizers and schedulers was set up for each fold, the training loop itself was initialized after setting the model in training mode. This loop iterated through the specified batches in the training dataloader and performed the required training steps. During this loop, the training loss, the training accuracy and, batch accuracy was also computed and accumulated for each batch as well. Similarly, after the training, the validation loop was set-up by putting the model in evaluation mode first. The validation loop similarly iterated through batches in the validation dataloader and performed the validation steps. This loop also computed and accumulated the validation loss, validation accuracy for each of the batches. And, after each fold ended, a confusion matrix was calculated and printed for the last epoch of that fold. Finally, metrics like accuracy, precision, recall, and F1 score were collected and calculated to evaluate the model's performance for each of the mentioned methods explained in the following subsections.

3.1.3 Full-Frozen Fine-Tuned BERT Model Implementation

The Full-Frozen model leverages the weights of the pre-trained BERT model without any further training. This model in our experimentation served as a baseline to understand the benefits of fine-tuning. This method is particularly useful in scenarios where rapid deployment is necessary, and computational resources are very constrained as well. This approach tests the model's capability to generalize based on its pre-existing knowledge. Since there is no training involved, the focus was particularly on evaluating the model's performance on the dataset within the classification layer. The metrics used here were same with the other methods as well for a consistent comparison with the other model implementations.

3.1.4 Half-Frozen Fine-Tuned BERT Model Implementation

In the Half-Frozen approach, only a subset of the BERT model layers are fine-tuned. In this case, a loop was initialized that freezed the first half of BERT's layers. This means these layers were not updated during training, and their pre-trained weights was retained as well. In a BERT model, as the layers are arranged such that the lower layers generally capture more basic aspects of language, while the upper layers are more involved in capturing complex language features, so that's why by freezing the lower layers and keeping the upper layers trainable, the main focus was on training the complex aspects of language processing and fine-tuning further on the specific dataset within the classification layer. As partial freezing reduces the computational cost and the risk of overfitting as well, it was a strategic choice to employ this method as the dataset itself was not large enough and the computational resources were limited as well.

3.1.5 Non-Frozen BERT Model Implementation

The Non-Frozen model uses all layers of the pre-trained BERT model. So, it allows the model to adapt as much as possible to the specific nuances of the dataset. By training all 12 layers, the model is fine-tuned to adapt to the specific linguistic characteristics and patterns present in the scientific abstracts and this comprehensive training allowed the model to modify both its basic language understanding and task-specific features as well. While it is mostly beneficial when the dataset is sufficiently large and diverse, but even with the small dataset in this case, this training process was still employed to further contrast and compare the differences of different tunings of the BERT model and it's impact on the overall model accuracy.

4 Results and Analysis

The comprehensive evaluation of the three fine-tuning strategies applied to the Full-Frozen, Half-Frozen, and Non-Frozen BERT models reveal results of model performance across various metrics. This section analyzes the experimental findings, providing a thorough comparison that illuminates the strengths and limitations of each approach in the context of scientific abstract classification.

4.1 Full-Frozen Fine-Tuned BERT Model Results

After being trained and tested, the evaluation of methods are conducted to understand how it performs in predicting abstracts across three different categories. Below are two tables that demonstrates this method’s model evaluation.

Table 2: Classification Report of Full-Frozen Fine-Tuned BERT Model

Category	Precision	Recall	F1 Score	Support
Physics	0.45	0.38	0.41	395
Medicine	0.34	0.10	0.15	395
Cybersecurity	0.39	0.73	0.51	395

Table 3: Overall Accuracy of Full-Frozen Fine-Tuned BERT Model

Overall Accuracy	Support
0.40	1185

Tables 2 and 3, reveals the performance metrics of the Full-Frozen Fine-Tuned BERT model across three distinct categories of Physics, Medicine, and Cybersecurity. While this method of training the BERT model demonstrated a moderate level of precision and recall in categories like Cybersecurity and Physics, it showed a notably lower performance in Medicine. The F1 score which balances the precision and recall, mirrored these patterns as well. The overall accuracy of the model stands at 0.40, reflecting a balanced but not high enough accuracy across a total of 1185 instances. As three categories were used for this training, this method proved that the achived accuracy was only a little bit higher than random guessing (0.33).

A stable but distinct pattern was also observed in its training and validation accuracies and losses as shown by the Figures 1 and 2 as well, with validation accuracy consistently lower than training accuracy across all folds. This trend hinted at a model that, while being robust to overfitting to a degree, it didn't fully capitalize on the hidden semantic patterns within the data due to its frozen state.

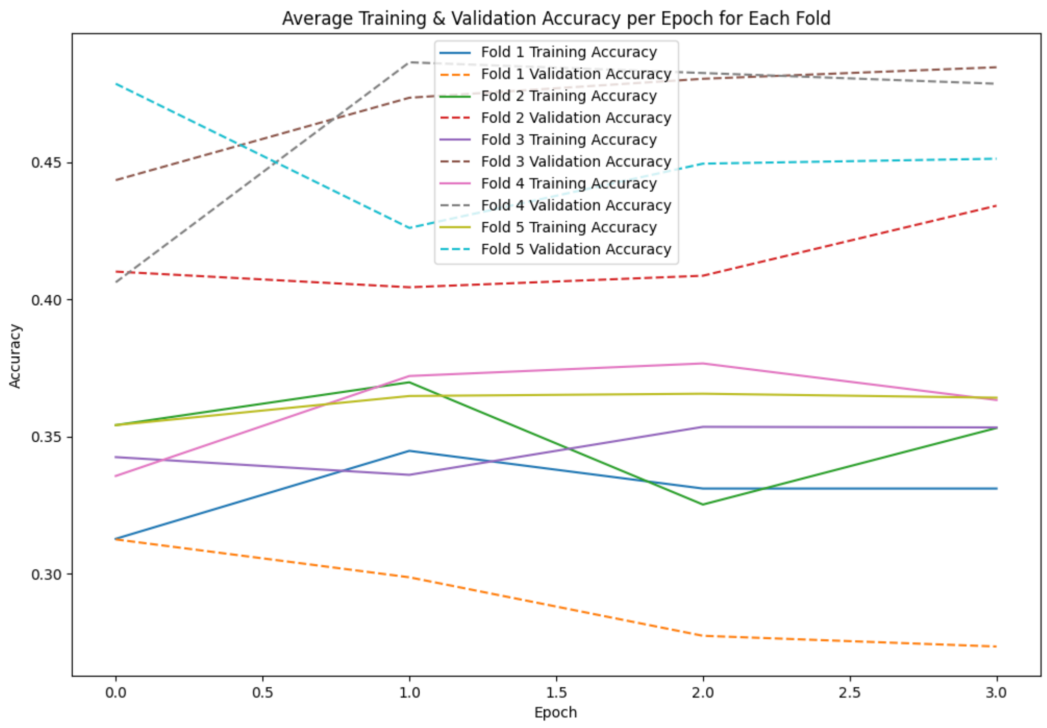

Figure 1: Average Training vs Validation Accuracy plot
for Full-Frozen Fine-Tuned BERT Model

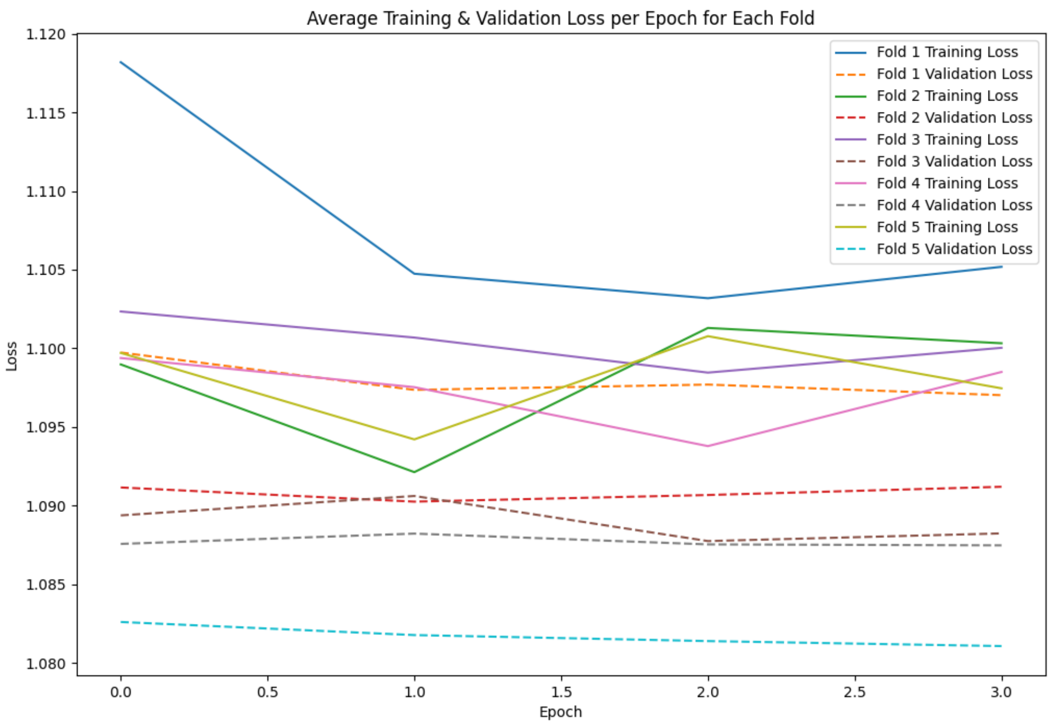

Figure 2: Average Training vs Validation Loss plot
for Full-Frozen Fine-Tuned BERT Model

The analysis of the confusion matrix in the Figure 3 also indicates a specific tendency towards the Cybersecurity category, with significant misclassifications, especially where abstracts from Physics were mistaken as Cybersecurity. Such a pattern reflects the model's reliance on the pre-trained weights, which in this case, did not align perfectly with the domain-specific language of the dataset.

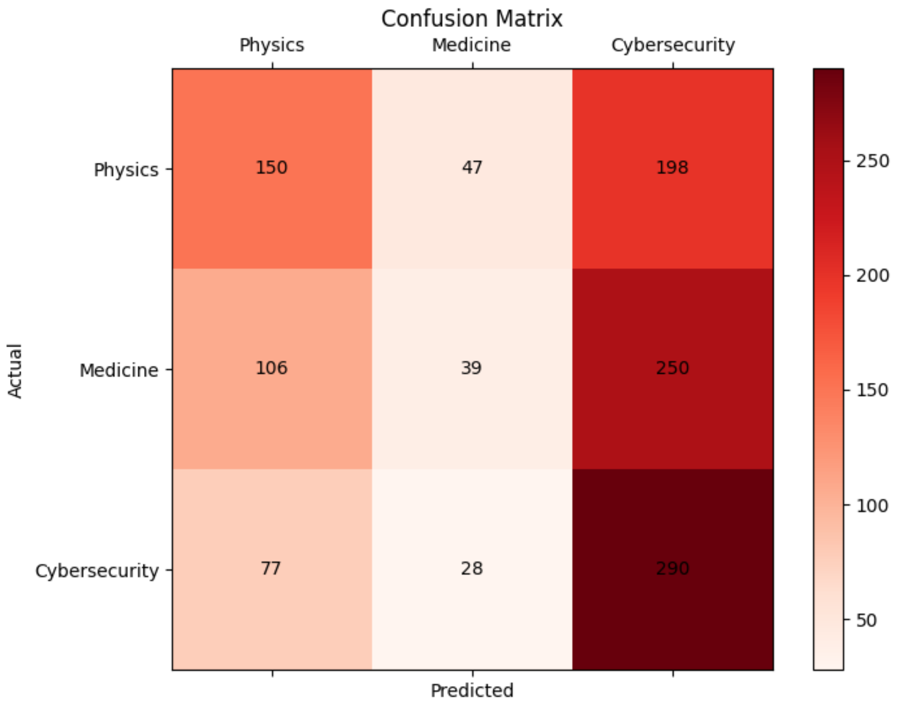

Figure 3: Confusion Matrix for Full-Frozen Fine-Tuned BERT Model

This similar finding is also observed when calculating the Accuracy from the confusion matrix as well.

$$Accuracy : \frac{TP_{sum}}{Total\ Predictions} \approx 0.40$$

In here, the TP_{sum} is the sum of the true positives for all classes which came out as 479 and $TotalPredictions$ is the total number of samples which is 1185.

4.2 Half-Frozen Fine-Tuned BERT Model Results

The Half-Frozen Fine-Tuned BERT model's training and validation performance was rigorously assessed to determine its effectiveness in classifying scientific abstracts as well.

Table 4: Classification Report of Half-Frozen Fine-Tuned BERT Model

Category	Precision	Recall	F1 Score	Support
Physics	0.97	0.98	0.97	395
Medicine	0.97	0.97	0.97	395
Cybersecurity	0.99	0.99	0.99	395

Table: 5: Overall Accuracy of Half-Frozen Fine-Tuned BERT Model

Overall Accuracy	Support
0.98	1185

Tables 4 and 5 clearly specify the performance metrics of the Half-Frozen Fine-Tuned BERT model. The precision and recall metrics are particularly high which indicates the this model's precision at correctly identifying and classifying abstracts into their rightful categories. The F1 score also reflect this high level of accuracy across all categories as well. The model achieved an overall accuracy of 0.98, which considerably surpassesd the baseline performance established by random guessing at 0.33.

The training and validation accuracies, as illustrated in Figures 4 and 5, also displayed a consistent parallelism across all folds. In the figures it is observed that the training accuracy closely mirrored the validation accuracy. This pattern suggested that the model was well-tuned to the data and was capable of learning and generalizing effectively without the common problem of overfitting as well.

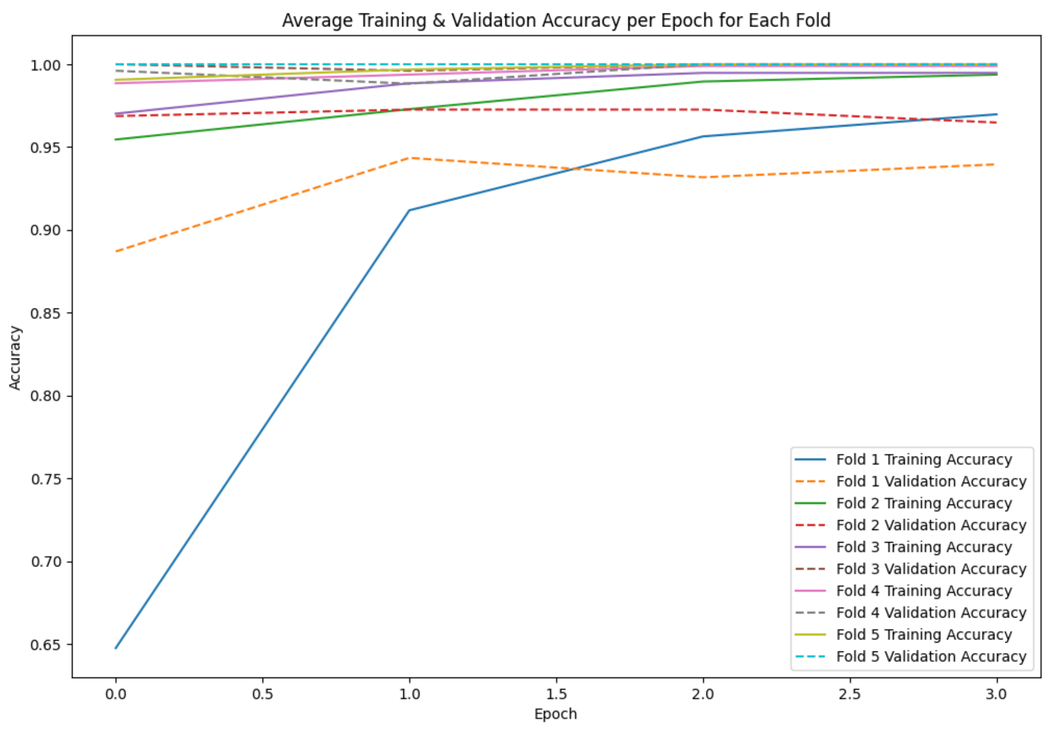

Figure 4: Average Training vs Validation Accuracy plot

for Half-Frozen Fine-Tuned BERT Model

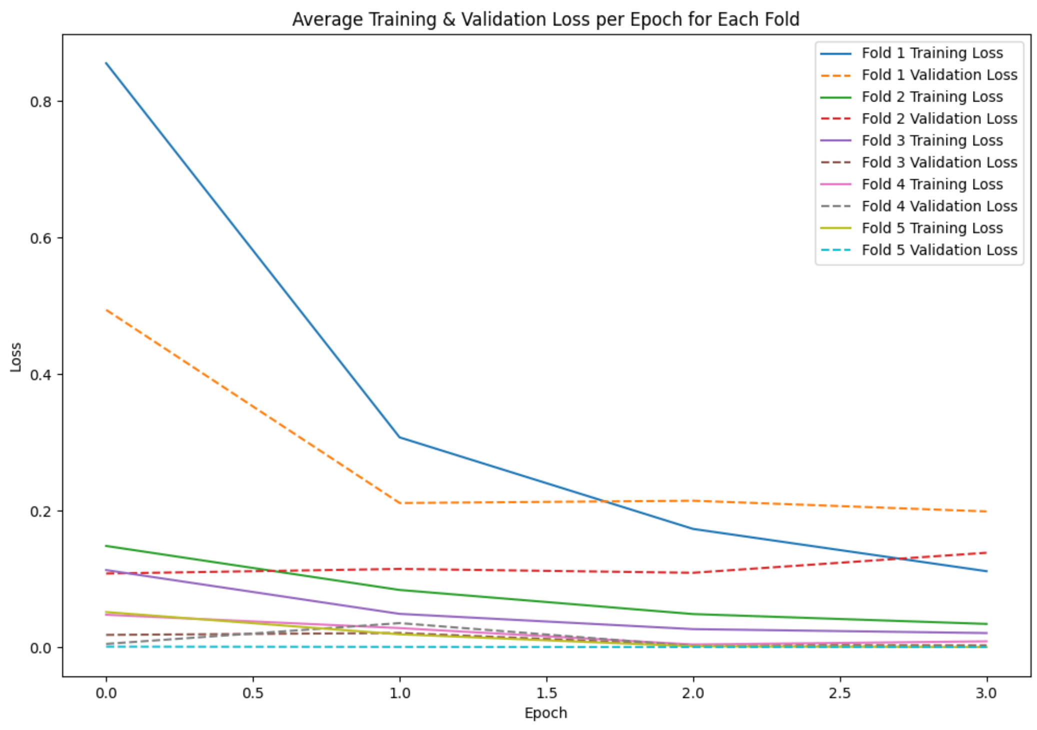

Figure 5: Average Training vs Validation Loss plot

for Half-Frozen Fine-Tuned BERT Model

The confusion matrix as depicted in Figure 6 also reinforces this method of model-tuning's robust classification capability. There is a strikingly low number of misclassifications, denoting a model that is not only well-adapted to the data, but also highly discriminative in its categorization.

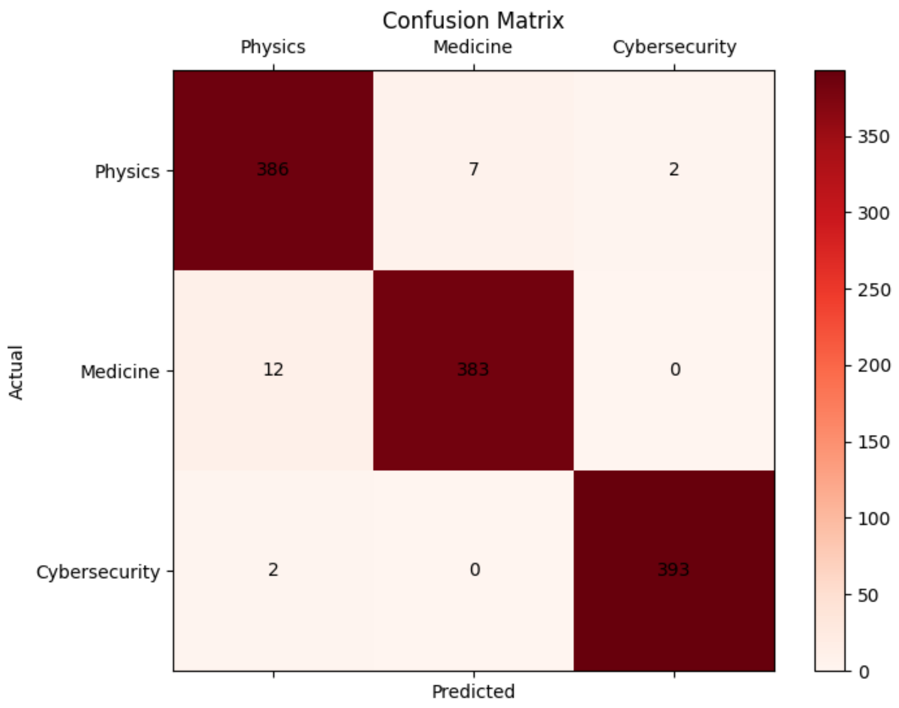

Figure 6: Confusion Matrix for Half-Frozen Fine-Tuned BERT Model

The similar trend is also observed in calculating the Accuracy from this method's confusion matrix as well.

$$Accuracy : \frac{TP_{sum}}{Total\ Predictions} \approx 0.98$$

In here, the TP_{sum} is similarly the sum of the true positives for all classes which came out as 1162 in this case and $TotalPredictions$ is the total number of samples (1185).

4.3 Non-Frozen BERT Model Results

The Non-Frozen BERT model underwent rigorous evaluation as well to understand its performance in classifying abstracts across the categories. The assessment of the model's effectiveness is captured in the performance metrics detailed below.

Table 6: Classification Report of Non-Frozen BERT Model

Category	Precision	Recall	F1 Score	Support
Physics	0.97	0.97	0.97	395
Medicine	0.96	0.98	0.97	395
Cybersecurity	0.99	0.97	0.98	395

Table 7: Overall Accuracy of Non-Frozen BERT Model

Overall Accuracy	Support
0.97	1185

In the above Table 6 and 7, we observe that the Non-Frozen BERT model exhibits exceptional precision, recall, and F1 scores across all categories as well. Notably, it achieves almost perfect metrics in the field of Cybersecurity, which suggests an overwhelming level of model training and an inherent understanding of the cybersecurity abstracts within the dataset. The overall accuracy of the model is at an impressive 0.97 which is a little bit lower than half-freezed training the BERT model. This result is also significantly higher than the baseline of random guessing as well which underscored the model's refined ability to discern and classify abstracts correctly.

The training and validation accuracies presented in Figures 7 and 8 reveal a consistent performance across all folds, with the validation accuracy closely tracking the training accuracy. This consistency is indicative of a well-fitted model that generalizes well to unseen data while avoiding overfitting.

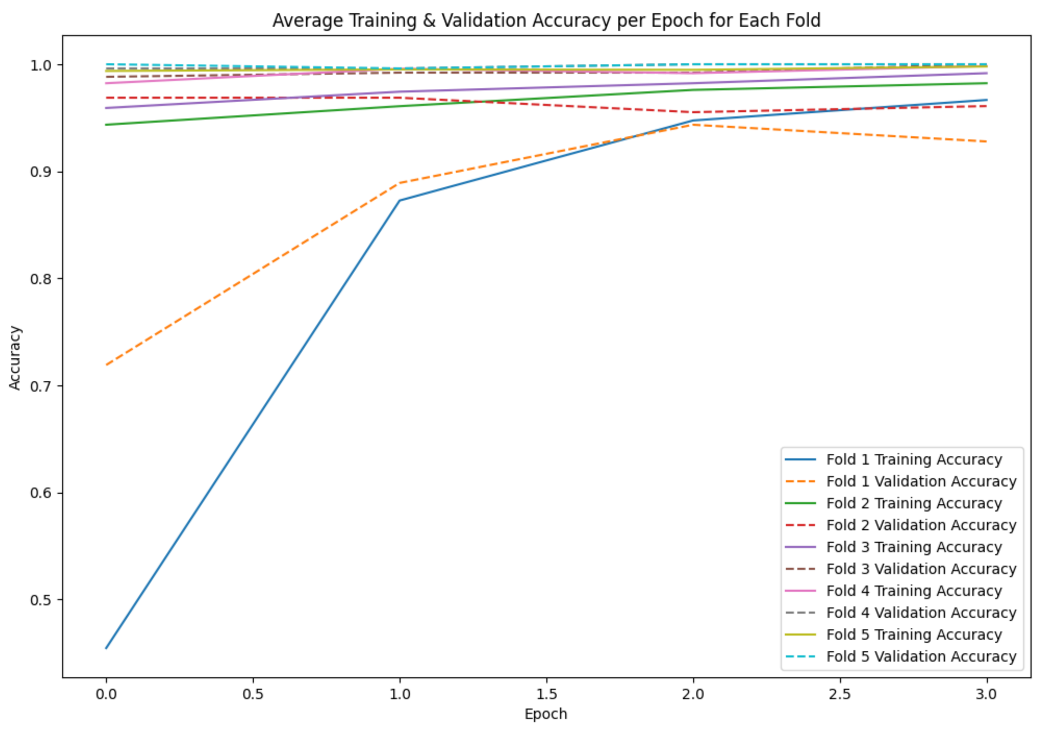

Figure 7: Average Training vs Validation Accuracy plot

for Non-Frozen BERT Model

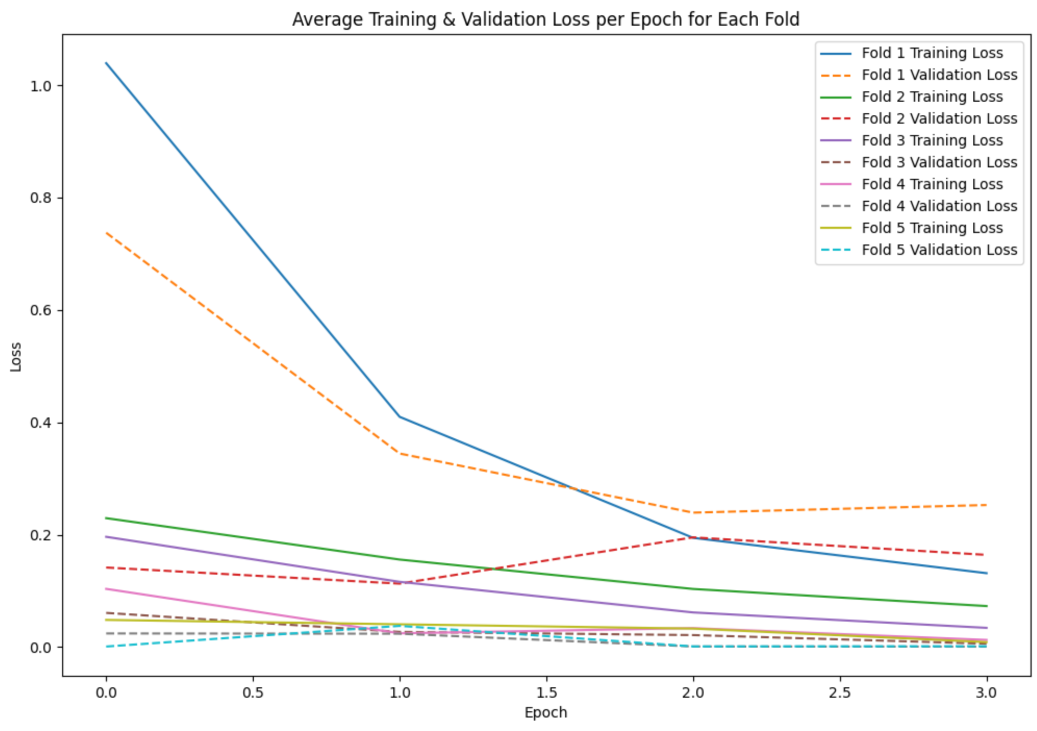

Figure 8: Average Training vs Validation Loss plot

for Non-Frozen BERT Model

Analyzing the confusion matrix in Figure 9 further solidifies the model's proficiency. It shows minimal confusion between categories, with the fewest misclassifications occurring between Physics and Cybersecurity. This high level of distinction between categories emphasizes the model's ability to leverage the full training process effectively, adapting the pre-trained BERT's weights to match the specialized content of the dataset.

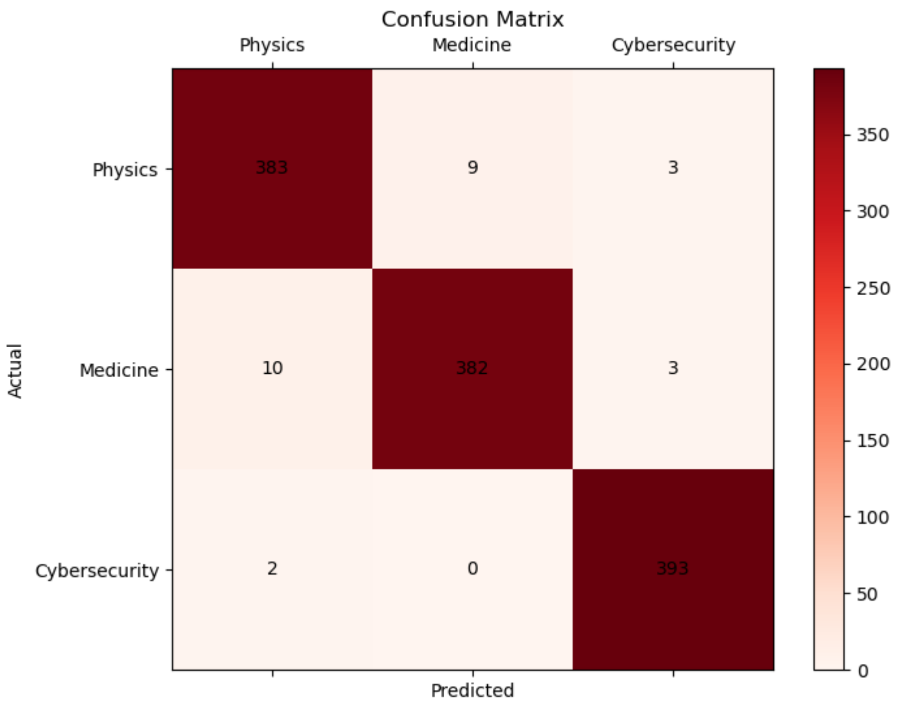

Figure 9: Confusion Matrix for Non-Frozen BERT Model

The accuracy that is calculated from this confusion matrix also followed the similar trend.

$$Accuracy : \frac{TP_{sum}}{Total\ Predictions} \approx 0.97$$

In here, the TP_{sum} is the sum of the true positives for all classes which came out as 1158 and $TotalPredictions$ is 1185 is this case as well.

5 Discussion

Firstly, using the Full-Frozen BERT model and relying exclusively on pre-trained parameters demonstrated moderate success. It particularly excels in classifying abstracts within the Cybersecurity domain, likely due to the presence of distinct, domain-specific terminology that the pre-trained BERT can recognize. However, the model's limitations are noticable in its subpar performance in Medicine, which is a domain where contextual understanding play a pivotal role. This suggested that while pre-trained knowledge is beneficial, it may not suffice for domains where context is intricately tied to specialized knowledge. The moderate overall accuracy of 0.40 which is only marginally above random chance (0.33), reinforced the notion that fully frozen layers may hinder the model's adaptability to the dataset's unique characteristics.

In contrast, the Half-Frozen BERT model presented a remarkable improvement, with high precision and recall across all domains. This model's strategy of fine-tuning only the latter layers allowed it to leverage both the general language understanding of the pre-trained BERT and the nuanced recognition of domain-specific features as well. The near-perfect overall accuracy of 0.98 underscored the effectiveness of this approach which suggested that allowing some degree of adaptability within the model's architecture significantly enhances its predictive capabilities as well.

Finally, the Non-Frozen BERT model showcased a slight decrement in overall accuracy compared to the Half-Frozen model while also taking more training time as well. Scoring a 0.97 accuracy, this method's performance still remained good, with high metrics across the board as well. The full fine-tuning process allowed the model to thoroughly learn the unique characteristics of the dataset, which was shown by the very few mistakes it made in distinguishing between different categories. This method's slight underperformance and more training time compared to the Half-Frozen model, could be attributed to overfitting, where the model becomes so specialized to the training data that its generalization to validation data slightly diminishes.

6 Conclusion

The comparative analysis makes it clear that the best way to adjust the model depends on the details of the data and what is needed to do for the given task. For sorting scientific abstracts, the Half-Frozen Fine-Tuned BERT model showed a balance between not needing too much computational power and still working really well at the same time as well. So, this method proved to be the best option for the task of scientific abstract classification, while also paving the way for more strategic and resource-conscious applications of the BERT model in future text classification tasks as well.