Domain Adaptation using Parameter Efficient Fine Tuning (PEFT)

Julian Roddeman (s1080491)
Radboud University
Nijmegen, Netherlands
julian.roddeman@ru.nl

Janneke Nouwen (s1101750)
Radboud University
Nijmegen, Netherlands
janneke.nouwen@ru.nl

Daan Brugmans (s1080742)
Radboud University
Nijmegen, Netherlands
daan.brugmans@ru.nl

1 INTRODUCTION

The integration of models like Bidirectional Encoder Representations from Transformers (BERT) in information retrieval systems has revolutionized the field of artificial intelligence and natural language processing (NLP) [2]. The dynamic nature of data domains presents a fundamental challenge: the adaptability of these models to previously unexplored domains. This research focuses on enhancing BERT's domain adaptability using Parameter Efficient Fine Tuning (PEFT), a method that fine-tunes a small set of parameters, thereby reducing computational costs and the risk of overfitting [8].

BERT's NLP ability is well established, but its performance in domains that it has not seen before, such as the data emerging from the COVID-19 pandemic, poses significant challenges [9]. BERT models pre-trained on extensive, yet specific, corpora may struggle to adapt to new, rapidly evolving datasets. The pandemic exemplifies this, presenting an abundance of novel information that was not part of BERT's original training data.

Traditional fine-tuning of models like BERT for new domains involves adjusting all of its 110 million parameters, a process that is computationally intensive and susceptible to overfitting [1]. PEFT offers a more efficient alternative by modifying only a subset of parameters, significantly reducing computational and storage demands, and making it more suitable for environments with limited resources.

This study examines the application of PEFT methods in adapting BERT for classifying document-query pairs related to COVID-19, a domain to which the original model was not exposed. The distinctive nature of the pandemic and the surge in specific literature make this domain an ideal testbed for evaluating PEFT methods in enhancing BERT's performance in classifying document-query relevancy from unseen domains. Our approach utilizes an optimized version of BERT, already fine-tuned for predicting relevancy judgement scores based on document-query data, since plain BERT is not capable of performing this task. This decision aligns with the need for more refined and domain-specific information retrieval post-pandemic. We aim to compare the enhanced domain adaptability of this optimized BERT model against its PEFT counterpart.

By proposing a more efficient fine-tuning approach, this research contributes to the field of information retrieval and has broader implications for the use of language models in rapidly evolving domains. Demonstrating the effectiveness of PEFT in adapting BERT to the COVID-19 domain lays the groundwork for future research into the agile adaptation of pre-trained models to unseen domains.

IR'23-'24, Radboud University, Nijmegen, Netherlands
2023.

2 RELATED WORK

The related literature emphasizes the difficulties and advances in adapting BERT-based models to evolving data domains. Thakur et al. [9] identify the limitations of BERT models in unseen domains, particularly in how they handle novel data such as information emerging from the COVID-19 pandemic. Addressing these challenges, PEFT, as discussed in the broader context of efficient model adaptation by Soekhoe et al. [8], presents an effective strategy for adapting BERT for specific tasks with reduced computational overhead and minimized risk of overfitting [1]. Additionally, the concept of using optimized models for better performance in specific tasks, similar to the use of adapter modules in NLP as proposed by Pfeiffer et al. [6], aligns with the need for efficient adaptation in resource-limited scenarios. This approach is particularly relevant in the context of the COVID-19 pandemic, where rapid and efficient information retrieval is crucial [10].

In our work, the PEFT method Low Rank Adaption (LoRA) as described in the work of Hu et al. [3] will be applied. LoRA encompasses the use of a new, additional weight matrix during model fine-tuning. The pre-trained weight matrix of the model is frozen, and only the new matrix is updated during fine-tuning, as shown in Figure 1. This new matrix may be vastly smaller than the original weight matrix, making it less computationally expensive to train and less prone to overfitting. This matrix is created by multiplying two matrices: A and B. A is randomly initialized at the start of fine-tuning. All values of B are set to 0. Because of this, at the start of the fine-tuning, the multiplication of these two matrices results in a null matrix. The randomly initialized values in A have therefore no influence of the performance of the model at the start of fine-tuning. After updating the weights, the pre-trained weight matrix and the newly trained weight matrix are then summed coordinate-wise with the original weights of the model. This is made possible by the parameter matrix of the original model having dimensions $d * k$, LoRA's matrix A having dimensions $d * r$ and matrix B having dimensions $r * k$, where r is the rank, a hyperparameter used when performing LoRA. A different hyperparameter α scales the weights of the matrix.

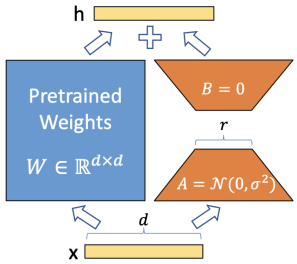

Figure 1: Abstraction of LoRA from Hu et al. [3].

3 METHODOLOGY

In this study, the performance of a pre-optimized BERT model versus this BERT model fine-tuned using PEFT in relevancy labelling COVID-19 related document-query pairs is compared. This comparison is performed using existing data and an existing BERT model pre-trained on predicting relevancy judgement scores of document-query pairs. This pre-trained model is a Sentence-BERT model taken from Reimers and Gurevych [7]. The specifically concerns the "msmarco-bert-base-dot-v5" model, a BERT model trained on data from the MS MARCO Passage Ranking dataset, optimized for representing documents as embeddings. This model, alongside other Sentence-BERT models, are hosted online on Hugging Face under the Sentence Transformers package. This package is also available as a Python package.

The dataset used to fine-tune the BERT and PEFT-BERT models is provided by Voorhees et al. [10] and is called the TREC-COVID dataset. The TREC-COVID dataset is a collection of documents, queries, and relevancy judgements on the topic of COVID-19 and the pandemic that followed it. The document collection consists of publications made during to the COVID-19 pandemic that are about COVID-19 and prior publications about other corona viruses. The collection of queries was formulated by the bio-medically trained organizers of the TREC-COVID dataset. The queries included in the dataset were inspired by common questions that were asked on social media. The collection of relevancy judgements was made by a group of medical students from the Oregon Health and Science University and the organizers of TREC-COVID. Professional indexers from the National Library of Medicine then assessed these relevancy judgements. For the relevancy judgements, there are 3 levels of relevancy: "Not Relevant", "Partially Relevant", and "Relevant". The TREC-COVID dataset was incorporated into the BEIR collection of IR evaluation datasets by Thakur et al. [9]. The version of TREC-COVID incorporated into BEIR was used for our research. The BEIR collection is hosted on Hugging Face.

Both the BERT and PEFT BERT models were trained and evaluated on the TREC-COVID dataset. In order for the pre-trained model to predict relevancy judgement scores, a dense linear layer was added at the end of the model's architecture that maps the pre-trained model's embeddings to the relevancy judgement scores. This final dense linear layer was then trained. While the additional dense linear layer was trained, the PEFT BERT model was further fine-tuned using LoRA, while the BERT model was not. Both models were then evaluated on data from the TREC-COVID dataset: given a document and a query, the model should predict the relevancy judgement score of the document-query pair.

4 EXPERIMENTAL SETUP

Our approach was implemented using the Python programming language. The SentenceTransformers [7] package was used to retrieve the pre-trained BERT model. The PyTorch and SentenceTransformer packages were used to alter and fine-tune the pre-trained BERT model. HuggingFace's Datasets [4] and PEFT [5] packages were used to retrieve the BEIR collection and to perform PEFT using LoRA respectively. Additionally, the TensorBoard [1] package allowed us to visualize the training process of the models. The code developed for our research can be found in a public GitHub repository (see appendix A).

[1] https://github.com/tensorflow/tensorboard

Table 1: Distribution of Relevancy Judgement Scores for the Unbalanced Pre-processed Document-Query Dataset

Relevancy Score	Relevancy Level	Count	Share
0	Not Relevant	117.346	62.52%
1	Partially Relevant	29.798	15.88%
2	Relevant	40.545	21.6%

The TREC-COVID dataset was extracted from the BEIR collection. Features from the TREC-COVID dataset besides the document titles, documents, queries, and relevancy judgement scores were dropped. This resulted in a dataset of document-query pairs and their relevancy judgement score. A total of 187.695 document-query pairs are included in this dataset. An overview of the data distribution of the dataset is given in table 1.

In addition to the document itself, BEIR's TREC-COVID dataset contains a title for every document. If such a title was present, we prepended it to the document string and processed this concatenation as the document. We dropped data points with missing documents, queries, and/or relevancy judgement scores. Invalid relevancy judgement scores, like -1, which in total occurred 3 times, were also excluded.

After this pre-processing, the dataset was split into train, validation, and test sets. We assumed a split of 60% train, 20% validation, and 20% test. The data was randomly shuffled prior to splitting. We used a random state for PyTorch, NumPy and Pythons internal Random module to ensure every model was trained with the same samples. Because the document-query pairs are are related in a many-to-many relationship, we applied extra pre-processing steps. Firstly, this involved making sets of all distinct document-ids and query-ids, and then allocate them in the train, validation and tests sets accordingly. Secondly, and extra check is done in the pre-processing pipeline to make sure no queries or documents overlap between sets. The downside of this pre-processing step, is that many document-query pairs in the dataset had to be dropped, decreasing the amount of usable data substantially. Table 2 shows the amount of document-query pairs in each split set.

As table 1 and table 2 show, the pre-processed document-query dataset was quite unevenly distributed. We expected that this may have a negative effect on the ability of a model to train on the data. Because of this, we implemented a data rebalancing procedure that attempts to balance the distribution of relevancy judgement scores across the train, validation, and test sets, while adhering to the rule that no document or query data must be leaked to another set. This was done by identifying the class distribution for all 3 sets, calculating where in these sets certain classes are over- or underrepresented, and moving a set of document-query pairs from one set to another. We used this strategy iteratively because moving document-query pairs from a set can free up space for another document-query pair to move. Every iteration, some entries of overbalanced classes in sets that contain most of the overbalanced class, are deducted to free up space for class balancing. This class balancing procedure resulted in train, validation, and test

Table 2: Distribution of Relevancy Judgement Scores for the unbalanced train, validation and test set

Split	Relevancy Score	Count	Share
Train	0	41.270	63.5%
	1	9.486	14.6%
	2	14.238	21.9%
	Total	**64.994**	**100%**
Validation	0	5.166	63.8%
	1	1.186	14.6%
	2	1.751	21.6%
	Total	**8.103**	**100%**
Test	0	4.705	60.2%
	1	1.427	18.3%
	2	1.685	21.6%
	Total	**7.817**	**100%**

Table 3: Distribution of Relevancy Judgement Scores for the balanced train, validation and test set

Split	Relevancy Score	Count	Share
Train	0	10.153	34.8%
	1	9.706	33.3%
	2	9.277	31.8%
	Total	**29.136**	**100%**
Validation	0	1.068	33.3%
	1	1.069	33.3%
	2	1.069	33.3%
	Total	**3.206**	**100%**
Test	0	940	37.9%
	1	712	28.7%
	2	828	33.4%
	Total	**2.480**	**100%**

sets with a more even class distribution. The consequence of class balancing is that the dataset sizes are further reduced. The number of document-query pairs in this balanced split are shown in table 3

A variety of hyperparameters were used in the training of BERT and PEFT BERT, specific combinations of which we tried out are given in table 1 and 4. We used learning rates of 1e-4 and 1e-5 alongside a Softmax Loss function. We trained models for 1 to 2 epochs on an unbalanced dataset, while training for 1 to 5 epochs on a balanced dataset. We applied a dropout of 10% during training on the final dense layer. All models were trained with a batch size of 32. Hyperparameters for LoRA include r and α. They were both set to 4 or 16, where r stands for the rank of LoRA's A and B matrices and α was set to the same value as the rank. Two scenarios were evaluated: one without class balancing, and one with class balancing which has fewer samples.

Model performance was evaluated using confusion matrices that describe the classification accuracy across the different relevancy judgement scores. The F1-score was calculated for each class label to provide a weighted measure of precision and recall. The weighted precision and weighted recall were also provided separately. We also report the training time, and the amount of parameters fine-tuned during training, which provides insights into the computational efficiency of the models. To get insight into the efficiency of LoRA, we also fine-tune the full BERT model without any frozen trainable parameters. Because this costs a significant amount of time, we only train the BERT model with one set of parameters and with the balanced dataset, as the balanced dataset is significantly smaller. All training and evaluation was done using a Nvidia GeForce RTX 3060 GPU with 12 GB of memory.

5 RESULTS

In table 4, the evaluation metrics of the different fine-tuned models are shown. The best performing BERT models and the best performing PEFT BERT models are highlighted in bold. The best performing model is the model that was fine-tuned using PEFT, has a learning rate of 1e-4, and an r and α of 16. We report a weighted F1-score of 0.520. This model was trained without balanced classes in the training set. The best BERT model without using PEFT trained without balanced classes has an F1-score of 0.305, which is substantially lower. When using balanced classes, the best performing BERT model without using PEFT has a F1-score of 0.367. This F1-score is lower than the F1-score of the PEFT model trained on the same balanced data: 0.389.

Validation and train scores can be found in table 1 and table 2 of appendix B. When comparing the test scores of the 2 best BERT models and the 2 best PEFT models to their train scores, we notice some overfitting for all four models. The model that shows the biggest difference between the train and test scores is the BERT model that was trained with balanced classes. Here we see that the train F1-score is 0.103 higher than the test F1-score. We notice that both models trained with balanced classes show a somewhat bigger difference between the train and test scores. This could be explained by the fact that these models were trained with a significantly smaller dataset, therefore needing more epochs for training.

What is also clear from table 4 is that there is very little difference in training time between the BERT models and the PEFT BERT models. For both models, a very low number of parameters was trained. Therefore, the training times are much smaller compared to fine-tuning all trainable parameters of BERT. We see a large difference in training time between the PEFT BERT model and the fully fine-tuned BERT model. One epoch using PEFT on BERT takes about 20 minutes, while fine-tuning all parameters of BERT on the same dataset takes almost 8 hours.

Table 5: Confusion Matrices for (PEFT) BERT Models on Test Set

BERT Model

Actual \ Predicted	0	1	2	Share
0	10	0	930	37.5%
1	1	0	711	28.7%
2	2	0	826	33.4%

PEFT BERT Model

Actual \ Predicted	0	1	2	Share
0	4357	26	322	60.2%
1	1315	5	107	18.3%
2	1540	4	141	21.6%

In the provided confusion matrices (table 5), the PEFT BERT model outperforms the BERT model, particularly in distinguishing different relevancy judgement scores. The PEFT BERT model accurately predicts 4357 instances for relevancy label 0. Even though

Table 4: Summary of Configurations and Weighted Performance Metrics for Different BERT Model Variants in Experimental Runs on Test Set. BERT and PEFT-BERT models with highest evaluation metrics are in bold.

Model	Learning Rate	Epochs	LoRA Config	Frozen Params	Training time	Recall	Precision	F1-Score
BERT	1e-4	1	-	100%	1.219hr	0.194	0.567	0.289
BERT	1e-4	2	-	100%	3.153hr	0.196	0.567	0.291
BERT	**1e-5**	**1**	**-**	**100%**	**1.219hr**	**0.223**	**0.483**	**0.305**
BERT	1e-5	2	-	100%	2.551hr	0.217	0.442	0.291
PEFT BERT	1e-4	1	r: 16, α: 16	99.45%	1.278hr	0.452	0.429	0.440
PEFT BERT	1e-4	1	r: 4, α: 4	99.85%	1.277hr	0.423	0.408	0.415
PEFT BERT	**1e-4**	**2**	**r: 16, α: 16**	**99.45%**	**2.749 hr**	**0.576**	**0.443**	**0.501**
PEFT BERT	1e-4	2	r: 4, α: 4	99.85%	2.774hr	0.194	0.567	0.289
PEFT BERT	1e-5	1	r: 16, α: 16	99.45%	1.278hr	0.488	0.464	0.476
PEFT BERT	1e-5	1	r: 4, α: 4	99.85%	1.293hr	0.263	0.497	0.344
PEFT BERT	1e-5	2	r: 16, α: 16	99.45%	2.689hr	0.602	0.362	0.452
Runs with Class Balancing:								
BERT	**1e-5**	**2**	-	**100%**	**39.27min**	**0.337**	**0.403**	**0.367**
BERT	1e-5	5	-	100%	2.713hr	0.288	0.399	0.335
BERT	1e-5	1	-	0%	7.705hr	0.285	0.492	0.361
PEFT BERT	1e-5	2	r: 16, α: 16	99.45%	41.18min	0.288	0.399	0.335
PEFT BERT	**1e-5**	**5**	**r: 16, α: 16**	**99.45%**	**1.867hr**	**0.383**	**0.395**	**0.389**

it has a share of 60.2% for label 0, it is still far outperforming the BERT model, which only correctly identifies 10 instances (share 37.5%). Relevancy label 1 is difficult for both models, with the BERT model failing to correctly classify any instances in this category and the PEFT BERT model managing only 5 correct predictions, indicating some difficulty in this area. For relevancy label 2, the BERT model has a slightly higher accuracy with 826 correct predictions, compared to the PEFT BERT model's 141. However, we should be cautious of this finding, because the BERT model's dataset has a greater percentage of relevancy label 2, and BERT predicts relevancy label 2 when it is actually 0 or 1, in almost all cases.

6 DISCUSSION AND OUTLOOK

The results suggest, despite not implementing class balancing, the PEFT BERT model outperforms the BERT model significantly. This implies that model architecture and training parameters, or inherent efficiencies in the PEFT BERT model, could play a critical role in its elevated predictive accuracy.

In analyzing the results for the BERT and PEFT BERT models (table 5), two main limitations stand out due to the nature of the dataset and the models' behavior. First, there is a clear trend in which the BERT model mostly predicts one specific class - relevancy label 2. This could imply that the model is not able to generalize enough, causing it to be unable to accurately recognise the other relevancy judgement scores. The PEFT BERT model shows a similar tendency, although to a lesser extent, implying that it may generalize better. Second, the class balancing process in the dataset presents a difficult situation. Because many document-query pairs in the original dataset overlap, removing duplicates to avoid repetition across training, testing, and validation sets results in significant data loss. This step could have an impact on the dataset's representativeness. In most real-world scenarios, the majority of documents are irrelevant (label 0), making this imbalance appear less problematic. However, it is important to remember that, despite being uncommon, finding relevant documents is often more important. As shown in table 4, we chose to not choose either of these arguments, and set class balancing as a hyperparameter. Table 4 shows that introducing class balancing increases BERT's F1-score significantly. As previously stated, BERT assigns a single label to each entry. Because this version of BERT is trained on a balanced dataset, the model learned to generalise slightly better, more evenly balancing the recall and precision, causing the F1-score to rise. PEFT BERT, however, does not share this performance boost when trained on a balanced dataset. This could mean that the extra size of the unbalanced dataset is impactful on improving performance, compared to class balancing with fewer entries. As a result, we can conclude that using PEFT for BERT classifiers reduces the susceptibility to class imbalance.

The relevancy judgement score 1 appears to be a problem for both the BERT and PEFT BERT models. This score is between 0 and 2, making it similar to both other labels and thus more difficult to classify. Based on this result, we can conclude that the BERT and PEFT BERT models have not yet been sufficiently optimised to classify three labels. We can also conclude from the confusion matrices in table 5 that PEFT BERT might do better in binary classifications.

6.1 Related Work

Hu et al.[3] state that applying PEFT to BERT-based models with LoRA improves generalizability, accuracy, and resource efficiency. In table 4 we see similarities. The BERT model requires approximately 7.7 hours to train the full set of parameters, whereas PEFT BERT (r: 16 α: 16) reduces this to 21 minutes per epoch while increasing generalizability and the F1-score.

Pfeiffer et al. [6] state that LoRA is well suited for domain adaptation and quick task switching in BERT-based models. Despite

the fact that our results show that PEFT BERT significantly increases training time and improves performance when compared to a basic BERT-based model, these results do not compare to other domain adaptation methods. As a result, we are unable to validate the findings of Pfeiffer et al..

6.2 Future Work

Taking the results of our research into account, we suggest some points of potential further research. We chose to adapt a pre-trained BERT model to data of the COVID-19 pandemic. Further research may show whether PEFT can be used to fine-tune BERT to improve performance on other unseen domains. Apart from this, we have only considered the use of LoRA as a PEFT method. However, the application of other PEFT methods following similar methodologies might prove fruitful. For example, there exist other PEFT methods that build upon LoRA, such as AdaLoRA [11], which attempts to reduce required computation further by only fine-tuning weights that provide the most information. This method may increase computational efficiency even further. Additionally, future work could focus on a retrieval use-case instead of a classification use-case as researched in this paper.

6.3 Conclusion

We conclude that fine-tuning (PEFT) BERT to out-of-domain datasets is not an easy task. Both models, BERT and PEFT BERT, were not able to achieve significant performance. We can conclude however, that fine-tuning BERT using PEFT, specifically LoRA, performs better than standalone BERT with frozen parameters during training. Therefore, we recommend doing more research towards the use of LoRA for information retrieval.

REFERENCES

[1] Mohammad Mahdi Bejani and Mehdi Ghatee. 2021. A systematic review on overfitting control in shallow and deep neural networks. *Artificial Intelligence Review* (2021), 1–48. https://doi.org/10.1007/s10462-021-09975-1

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[3] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*. https://openreview.net/forum?id=nZeVKeeFYf9

[4] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A Community Library for Natural Language Processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 175–184. arXiv:2109.02846 [cs.CL] https://aclanthology.org/2021.emnlp-demo.21

[5] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods. https://github.com/huggingface/peft.

[6] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A Framework for Adapting Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 46–54. https://doi.org/10.18653/v1/2020.emnlp-demos.7

[7] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. https://arxiv.org/abs/1908.10084

[8] Deepak Soekhoe, Peter Putten, and Aske Plaat. 2016. On the Impact of Data Set Size in Transfer Learning Using Deep Neural Networks. 50–60. https://doi.org/10.1007/978-3-319-46349-0_5

[9] Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, J. Vanschoren and S. Yeung (Eds.), Vol. 1. Curran. https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/65b9eea6e1cc6bb9f0cd2a47751a186f-Paper-round2.pdf

[10] Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. TREC-COVID: Constructing a Pandemic Information Retrieval Test Collection. *SIGIR Forum* 54, 1, Article 1 (feb 2021), 12 pages. https://doi.org/10.1145/3451964.3451965

[11] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=lq62uWRJjiY

Appendices

A CODE

The following link leads to the GitHub repository containing the code used in this research: https://github.com/JulianRodd/projectInformationRetrieval

B EVALUATION METRICS

Table 1: Summary of Configurations and Weighted Performance Metrics for Best BERT Model Variants in Experimental Runs on Validation Sets

Model	Learning Rate	Epochs	LoRA Config	Recall	Precision	F1-Score
BERT	1e-5	1	-	0.223	0.462	0.301
PEFT BERT	1e-4	2	r: 16, α: 16	0.604	0.450	0.516
Runs with Class Balancing:						
BERT	1e-5	2	-	0.334	0.213	0.260
PEFT BERT	1e-5	5	r: 16, α: 16	0.337	0.377	0.356

Table 2: Summary of Configurations and Weighted Performance Metrics for Best BERT Model Variants in Experimental Runs on Train Sets

Model	Learning Rate	Epochs	LoRA Config	Recall	Precision	F1-Score
BERT	1e-5	1	-	0.225	0.491	0.309
PEFT BERT	1e-4	2	r: 16, α: 16	0.610	0.475	0.534
Runs with Class Balancing:						
BERT	1e-5	2	-	0.319	0.226	0.264
PEFT BERT	1e-5	5	r: 16, α: 16	0.346	0.324	0.335