Proceedings of CONF-MLA 2025 Symposium: Intelligent Systems and Automation: AI Models, IoT, and Robotic Algorithms
DOI: 10.54254/2755-2721/2025.LD27825

Optimizing BERT Fine-tuning Strategies: A Hyperparameter and Architecture Analysis for Sentence Pair Classification

Lu Liu

Sino-European School of Technology, Shanghai University, Shanghai, China
liulu@shu.edu.cn

Abstract. To address the challenges in fine-tuning Pre-trained Language Models (PLMs) like BERT, where performance is highly sensitive to architecture and hyperparameter choices, this study proposes and validates a systematic two-stage optimization process using the MRPC sentence pair classification task. Progressing from architecture exploration to parameter optimization, our experiments first reveal that a simplified single-layer linear classifier outperforms more complex structures for this task. Subsequently, large-scale hyperparameter tuning identifies batch size as the most critical parameter, while others like learning rate exhibit a distinct optimal range. By implementing this structured methodology, we significantly improved the model's validation accuracy. This work demonstrates that a methodical approach, combining fine-grained architecture adaptation with systematic parameter tuning, is crucial for realizing the full potential of pre-trained models.

Keywords: BERT, Model Fine-tuning, Hyperparameter Optimization, Classifier Architecture, Natural Language Processing.

1. Introduction

In recent years, the field of Natural Language Processing (NLP) has undergone a paradigm shift led by Pre-trained Language Models (PLMs). Models such as BERT (Bidirectional Encoder Representations from Transformers) [1] have acquired profound language knowledge and contextual understanding capabilities through self-supervised learning on massive unlabeled text corpora. Subsequently, by fine-tuning on labeled data for specific downstream tasks, these models can adapt to new tasks with high efficiency, achieving state-of-the-art results on various benchmarks, including text classification, semantic matching, and question answering, thereby becoming the mainstream technological paradigm in NLP [2, 3].

However, the immense success of this "pre-training and fine-tuning" paradigm has also obscured the complexity and challenges inherent in the fine-tuning process itself [4]. Fine-tuning is not a simple "plug-and-play" process; its final effectiveness is highly dependent on the choice of fine-tuning strategy and hyperparameter configuration. Research and practice have shown that inappropriate fine-tuning schemes, such as using general-purpose default parameters or arbitrarily constructing a task-specific head, often fail to fully unleash the potential of pre-trained models. This can even lead to unstable model training and poor generalization, especially on small to medium-sized datasets [5]. Therefore, how to systematically optimize the fine-tuning process has become a

DOI: 10.54254/2755-2721/2025.LD27825

crucial link in translating the powerful capabilities of pre-trained models into practical application value.

To address these challenges, this study aims to deeply investigate and validate two key optimization paths for the BERT fine-tuning process: model architecture optimization and training process optimization. Specifically, this paper proposes and seeks to answer the following two core questions:

The Impact of Classifier Structure: Given BERT's powerful feature extraction capabilities, does a deeper, more complex classifier always lead to better performance? Or, for a specific task, is a simplified structure more effective?

The Sensitivity and Interaction Effects of Hyperparameters: How do key hyperparameters such as learning rate, batch size, and weight decay individually and collectively affect model performance? Is there an optimal combination of parameters that can significantly enhance the final performance?

To answer these questions, this paper uses the MRPC (Microsoft Research Paraphrase Corpus) task from the GLUE benchmark as an experimental platform and designs a progressive two-stage optimization process. First, through comparative experiments, we explore the impact of different classifier complexities on model performance to identify an efficient baseline model architecture. Subsequently, building on this architecture, we conduct a large-scale global hyperparameter grid search to systematically analyze the independent and interactive effects of various parameters and identify the optimal training configuration.

The main contribution of this study is not only validating the effectiveness of simplifying classifier structure and systematic hyperparameter tuning in BERT fine-tuning but, more importantly, proposing and implementing a structured optimization process from "architecture exploration" to "parameter optimization." Experimental results show that this process significantly improves model performance, increasing the validation accuracy from an unoptimized baseline of approximately 79% to 84.52%. We hope that the findings and methods of this study will provide valuable practical guidance and theoretical reference for researchers and engineers in related fields when fine-tuning models.

The remainder of this paper is organized as follows: Chapter 2 reviews the background and related work on pre-trained language models and fine-tuning techniques. Chapter 3 details the experimental design, including the dataset, model, and the two-stage optimization strategy, and presents and analyzes the experimental results. Finally, Chapter 4 concludes the paper and discusses future research directions.

2. Research purpose and background survey

2.1. Research purpose

In the field of Natural Language Processing (NLP), pre-trained language models represented by BERT have made groundbreaking progress in numerous tasks through the "pre-train and fine-tune" paradigm [1]. However, fine-tuning is not a straightforward process; its final outcome heavily relies on the chosen fine-tuning strategy and hyperparameters [2]. An "out-of-the-box" or arbitrarily configured fine-tuning scheme often fails to fully leverage the potential of pre-trained models, especially on small to medium-sized datasets, where models are prone to overfitting and poor generalization.

Therefore, the core objective of this study is to systematically investigate and validate two key optimization paths for the BERT fine-tuning process to enhance its performance on specific

DOI: 10.54254/2755-2721/2025.LD27825

downstream tasks. Specifically, using the MRPC (Microsoft Research Paraphrase Corpus) task from the GLUE benchmark as a case study, this research unfolds in stages:

• Exploring Fine-tuning Architecture Optimization: This involves investigating the impact of different structural complexities of the top-level classifier on BERT's model performance. This study aims to answer a critical question: for a model like BERT, which already possesses powerful feature extraction capabilities, does a deeper, more complex classifier always yield better performance?

• Exploring Training Process Optimization: After identifying a superior classifier structure, this study will conduct a large-scale global hyperparameter tuning. We will systematically analyze the non-linear impact of key hyperparameters like learning rate and batch size, as well as their interactions, on model performance to find the optimal training configuration.

Through these experiments, this study aims to provide an effective, end-to-end optimization framework—from architecture to parameters—for fine-tuning BERT on specific NLP tasks, offering both practical guidance and insights into the mechanisms behind different optimization strategies.

2.2. Background survey

2.2.1. The rise of pre-trained language models

In recent years, with the introduction of the Transformer model, pre-trained language models have achieved unprecedented success [3]. BERT (Bidirectional Encoder Representations from Transformers) stands out as a prime example, learning deep language representations through pre-training on massive unlabeled text corpora with tasks like Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) [1]. This "pre-train and fine-tune" paradigm allows it to achieve state-of-the-art performance on various downstream tasks, such as text classification and named entity recognition, with only a small amount of labeled data.

2.2.2. The role and challenges of fine-tuning

Fine-tuning is a crucial step in adapting general-purpose pre-trained models to specific downstream tasks [2]. During this process, the model's parameters are updated based on the labeled data of the target task, enabling it to learn task-specific features and patterns. Studies have shown that fine-tuning primarily affects the higher layers of the model (especially the last few), adapting their attention patterns and feature representations to the downstream task, while the lower layers retain most of the general linguistic knowledge [6].

However, the fine-tuning process is not without its challenges. Two core challenges form the motivation for this study:

The Challenge of Architecture Selection: When fine-tuning, it is common to add one or more new "task-specific heads," such as a classifier, on top of the BERT model. The design of this task head (e.g., number of layers, use of activation functions or Dropout) directly affects the model's learning capacity and final performance. Designing a classifier that can effectively utilize BERT's features without introducing excessive redundant parameters is a question worth exploring.

The Challenge of Hyperparameter Sensitivity: Large models like BERT are highly sensitive to hyperparameters such as learning rate, batch size, and weight decay [5]. Improper hyperparameter settings can lead to unstable training, slow convergence, or even severe performance degradation. As pointed out by Liu and Wang, automated hyperparameter optimization (HPO) methods may not even outperform a simple grid search under a limited time budget, which highlights the complexity of the

DOI: 10.54254/2755-2721/2025.LD27825

tuning process [7]. Due to the complex nonlinear relationships between hyperparameters, finding the optimal combination usually requires systematic experimental search.

2.2.3. Fine-tuning optimization methods and related work

In response to the above challenges, the academic community has conducted extensive exploration. In terms of architecture optimization, some studies have shown that for certain tasks, a more complex structure is not always better. For instance, Arase and Tsujii [8] found in their research that a "transfer fine-tuning" method that simplifies the feature generation process can achieve superior performance on small-scale datasets. Their conclusion—that "simple feature generation methods are more effective than complex ones"—provides important inspiration for this study's exploration of simplified classifier structures, suggesting that the features extracted by BERT may already be powerful enough, not requiring an overly complex classifier for further processing.

In terms of hyperparameter optimization, systematic tuning is considered a necessary means to improve model performance. Manual tuning is time-consuming and inefficient, so automated methods such as Grid Search, Random Search, and Bayesian Optimization are widely used [9]. Especially for large language models like BERT and the GPT series, their high training and inference costs, as well as their impact on energy consumption and the environment, have brought new challenges to the optimization process. This has spurred research into more economical and efficient hyperparameter optimization methods. For example, Wang et al. [10] proposed a framework called EcoOptiGen, designed specifically for the inference optimization of large language models. It systematically finds the optimal combination of inference hyperparameters within a limited budget by combining economical hyperparameter optimization strategies with cost-based pruning techniques. Sun et al. also confirmed that appropriate fine-tuning of BERT for specific tasks is key to achieving good classification results [11]. The in-depth analysis of BERT fine-tuning dynamics by Hao et al. [6] also indirectly corroborates the complexity of the influence of parameters like learning rate on different layers of the model, highlighting the importance of refined tuning.

2.2.4. The GLUE dataset and the MRPC task

To standardize the evaluation of the generalization ability of NLP models, the GLUE (General Language Understanding Evaluation) benchmark was proposed [12]. It covers nine tasks, including semantic similarity and natural language inference. The MRPC (Microsoft Research Paraphrase Corpus) task selected for this study is a classic binary classification task within it, which requires the model to determine whether two sentences are paraphrases of each other [13]. The MRPC dataset is of a moderate scale and has high requirements for the model's semantic understanding ability, making it an ideal experimental platform for validating the effectiveness of fine-tuning strategies.

2.2.5. Existing problems and summary

Although the BERT fine-tuning paradigm has become mainstream, how to efficiently and stably obtain optimal performance remains a practical challenge. Most existing research has focused on proposing new pre-training methods or model architectures, while the systematic optimization strategies for the fine-tuning process itself—particularly the relationship and practical workflow between classifier architecture selection and global hyperparameter tuning—have not been sufficiently discussed. Many practitioners tend to use recommended default parameters or conduct

DOI: 10.54254/2755-2721/2025.LD27825

small-scale trial-and-error, which limits the upper bound of the model's performance [14]. Therefore, this study aims to fill this gap by providing a more instructional optimization framework for the BERT fine-tuning process through systematic experiments.

3. Experiments and results analysis

To validate the effectiveness of the fine-tuned BERT model on a specific natural language processing task and to explore the impact of key hyperparameters and different fine-tuning strategies on model performance, this study designed and implemented the following experiments. This chapter will detail the experimental environment, dataset, model configuration, experimental design, and provide an in-depth analysis and discussion of the experimental results.

3.1. Experimental methods

The software environment for this experiment is based on the Python language, with PyTorch as the core framework. The construction, training, and fine-tuning of the model primarily rely on the Hugging Face Transformers library, while the Datasets library is used for loading and processing the dataset. All code is configured to prioritize CUDA acceleration to ensure training efficiency.

This study uses a public sentence-pair classification task dataset for the experiment. The task aims to determine the relationship between two input sentences (sentence1 and sentence2).

The data preprocessing pipeline is as follows:

• Tokenization: Use BertTokenizer to tokenize the input sentence pairs and convert them into input IDs acceptable to the model.

• Padding & Truncation: To handle text sequences of varying lengths, this study processes all input sequences to a uniform max_length=128. Sequences longer than this are truncated, and shorter ones are padded.

• Data Loading: Use PyTorch's DataLoader to construct a data pipeline for batch loading.

The experiment selects bert-base-uncased as the pre-trained model and builds a BertForSequenceClassification model on top of it to handle the downstream sentence-pair classification task. The model's optimizer is AdamW, used in conjunction with a linear learning rate warmup and decay strategy, which helps stabilize the model in the early stages of training and achieve better convergence later on.

The experiment in this study follows a progressive optimization process from coarse to fine, mainly consisting of two stages:

1. Fine-tuning Strategy Exploration: First, based on a fixed set of preliminary hyperparameters, this study explores the impact of classifiers with different complexities on model performance, aiming to determine an efficient base model architecture suitable for this task.

2. Global Hyperparameter Tuning: Based on the optimal classifier architecture identified in the first stage, this study conducts a large-scale global hyperparameter search to further squeeze model performance and find the optimal combination of training parameters.

The goal of this stage of the experiment is to evaluate the impact of different classifier network architectures on top of the BERT model on the final performance. This study starts with an initial version of a fine-tuned model and designs four different model variants for comparison by increasing network depth or simplifying the architecture.

Base Model (Finetune1): The classifier consists of a Dropout layer and a linear layer.

Deeper Models (Finetune2, Finetune3): On top of the base model, one and two additional combinations of a ReLU activation and a linear layer are added, respectively.

DOI: 10.54254/2755-2721/2025.LD27825

Simplified Model (Finetune4): The classifier of the base model is replaced with a simpler single-layer linear network.

After determining the optimal classifier architecture, this study uses it as the base model for comprehensive hyperparameter tuning. This experiment adopts a strategy combining control variables and grid search, systematically adjusting the following five key hyperparameters:

Learning Rate: [2e-5, 3e-5, 4e-5, 5e-5]

Batch Size: [4, 8, 16, 32, 64]

Weight Decay: [0.001, 0.01, 0.1]

Warmup Ratio: [0.05, 0.06, 0.07, 0.08]

Epochs: [4, 5]

The evaluation metric for the experiment is the accuracy on the validation set. After each training epoch, the model is evaluated on the validation set, and the model state with the highest validation accuracy is saved as the best model for the current parameter combination.

3.2. Results and discussion

The first step of this study was to determine an efficient classifier base architecture. Starting with an initial fine-tuned model, this study conducted a series of comparative experiments by adjusting the network structure of its top classifier.

• Initial Fine-tuned Model (Base): The starting point for this series of experiments was our initial fine-tuned model (Finetune1). This model was preliminarily adjusted based on the standard bert-base-uncased, with its classifier consisting of a Dropout layer and a linear layer. On the test set, this model achieved an accuracy of 83.13%, serving as the baseline for subsequent optimizations.

• Increasing Classifier Depth: Building on the initial fine-tuned model, this study added a ReLU activation function and a second linear layer after the original linear classification layer (Finetune2). This adjustment resulted in a slight increase in model accuracy to 83.19%. Subsequently, this study attempted to deepen it further by adding a third linear layer (Finetune3), but this time the model accuracy dropped to 83.07%.

• Simplifying the Classifier Architecture: Given that deepening the classifier network did not yield significant benefits, this study tried the opposite approach, replacing the classifier in the initial model with a new, simpler single-layer linear network (Finetune4). Unexpectedly, this simplification achieved the best result in this series of experiments, with the model accuracy increasing to 83.30%.

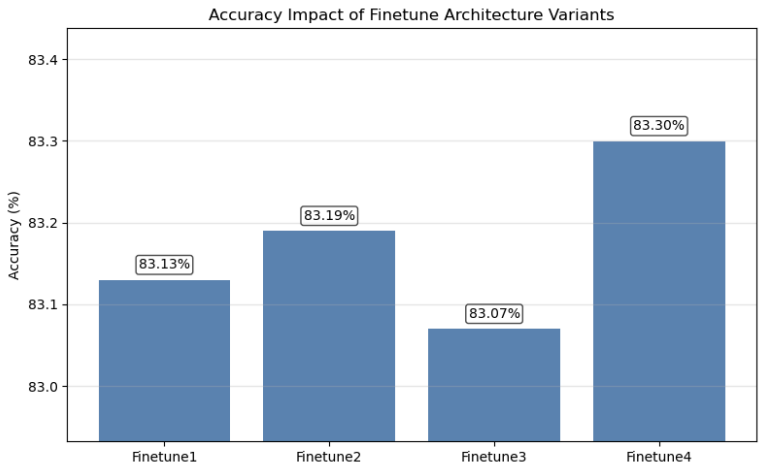

Figure 1. Impact of network architecture on model accuracy

DOI: 10.54254/2755-2721/2025.LD27825

This series of experimental results indicates that, for this study's specific sentence-pair classification task, the features extracted by the BERT encoder are already highly separable. In this situation, a complex, deep classifier is not only unnecessary but may also lead to overfitting or training instability due to the introduction of too many parameters, thereby harming the model's generalization ability. In contrast, a structurally simpler single-layer linear classifier can more directly and efficiently utilize BERT's powerful feature representations, ultimately achieving superior performance.

Therefore, this study decided to adopt this simplified single-layer linear classifier architecture as the base model for the subsequent global hyperparameter tuning.

After determining the optimal classifier architecture (i.e., the architecture of Finetune4), this study used it as a basis to conduct large-scale global hyperparameter tuning experiments, aiming to maximize the model's performance.

To intuitively analyze the impact of individual hyperparameters on model performance, this study plotted the relationship between each hyperparameter and model accuracy (as shown in Figure 2).

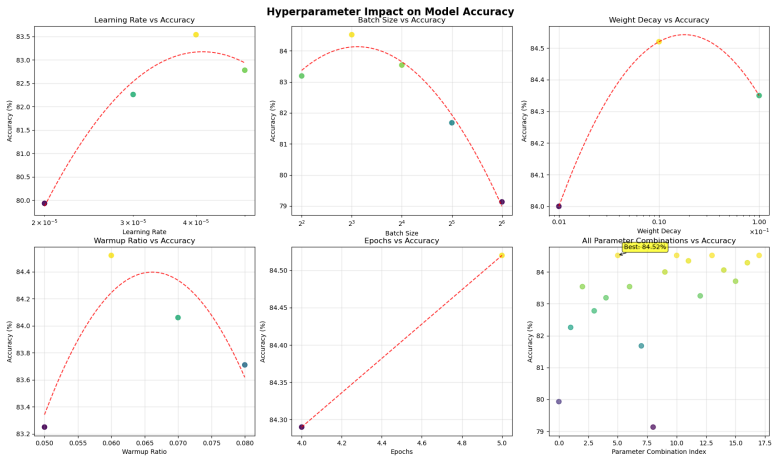

Figure 2. Impact of hyperparameters on model accuracy

From Figure 2, the following patterns can be observed:

1. Learning Rate: Model accuracy shows a trend of first rising and then falling as the learning rate increases, peaking around 4e-5. This indicates that a learning rate that is too low may lead to insufficient model convergence, while one that is too high may cause instability in the training process, thus harming performance.

2. Batch Size: Batch Size is the most significant parameter among all tested. Accuracy continuously improves as the Batch Size increases from 4 to 8, reaching its highest point at 8; however, as the Batch Size continues to increase, the accuracy drops. This suggests that an overly large Batch Size may lead to a decrease in the model's generalization ability.

3. Weight Decay and Warmup Ratio: These two parameters also exhibit a nonlinear "parabolic" relationship, achieving optimal performance around 0.01 and 0.06, respectively. This demonstrates that appropriate regularization and learning rate strategies are crucial for model convergence.

4. Epochs: Within the tested range of 4 to 5 epochs, increasing the number of training epochs steadily improves model accuracy, indicating that the model is still in a learning state at this stage and has not yet shown signs of overfitting. However, by observing the changes in the loss function during training, it was found that when the number of training epochs reached five, the loss function showed very little change (see Figure 3). It can be inferred from this that if the number of epochs

DOI: 10.54254/2755-2721/2025.LD27825

were increased to six, overfitting would be highly likely to occur. Therefore, keeping the number of training epochs at five should be the better choice.

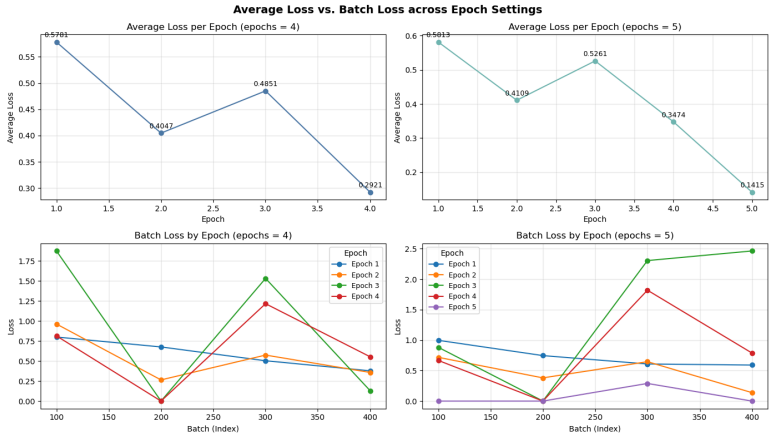

Figure 3. Change in average loss and batch loss over training epochs

The scatter plot in the bottom-right corner of Figure 2 displays the experimental results for all parameter combinations, with the optimal combination achieving a validation set accuracy of 84.52%.

To further quantify the relationships between variables from a statistical perspective, this study calculated and plotted a correlation matrix heatmap of the hyperparameters and model accuracy (as shown in Figure 4).

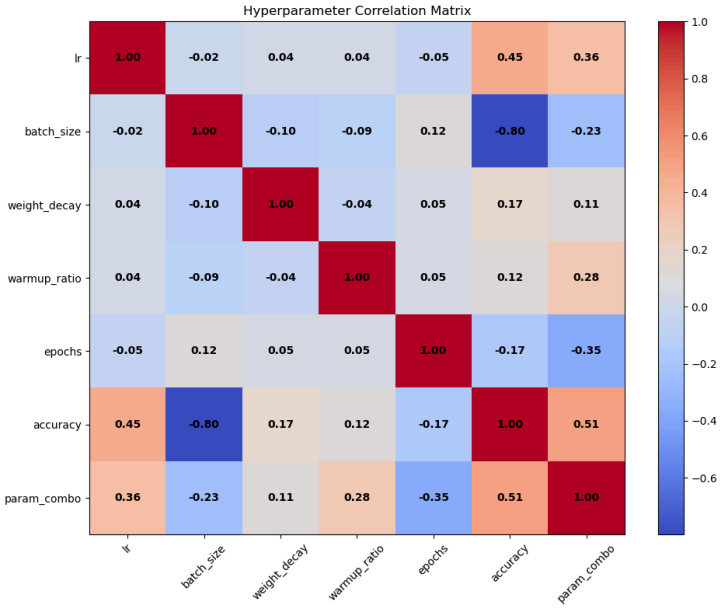

Figure 4. Hyperparameter correlation matrix

This matrix reveals deeper associations:

• Strongest Correlation: There is a strong negative correlation of -0.80 between batch_size and accuracy. This is perfectly consistent with the observations from Figure 2, once again confirming that the overall trend is that a larger Batch Size leads to lower model accuracy, highlighting its importance as a key hyperparameter.

DOI: 10.54254/2755-2721/2025.LD27825

• Moderate Correlation: There is a positive correlation of 0.45 between lr (learning rate) and accuracy. This reflects that within the parameter range of this experiment, increasing the learning rate generally had a positive effect, but this masks the nonlinear relationship observed in Figure 2.

• Weak Correlation and Potential Conflict: The correlation between epochs and accuracy is -0.17, showing a weak negative correlation. This contradicts the intuitive conclusion from Figure 2 that "more epochs lead to higher accuracy." This is likely due to a Confounding Effect among multiple variables—that is, some poorly performing parameter combinations (e.g., a very large Batch Size) may have happened to be set with more training epochs, thus statistically lowering the positive impact of epochs on a global scale. This phenomenon precisely illustrates that hyperparameter optimization is a complex nonlinear problem, and single-variable analysis and multi-variable correlation analysis must be interpreted together.

3.3. Chapter summary

This chapter, through a two-stage systematic experiment, has deeply investigated the impact of classifier architecture and key hyperparameters on the performance of a BERT fine-tuned model. The experimental results show that:

Exploration of the fine-tuning strategy is a crucial first step in optimization. The experiment first demonstrated that, for this task, a simpler single-layer linear classifier (test set accuracy of 83.30%) is superior to more complex multi-layer structures, establishing an efficient base architecture for subsequent optimization.

Batch Size is the most critical hyperparameter affecting model performance. On the optimal model architecture, this study found the best value to be 8; values that are too large or too small lead to a significant drop in performance.

Learning rate, weight decay, and warmup ratio all have an optimal range, and deviating from this range leads to a decrease in performance, validating the necessity of refined tuning.

Through the two-stage strategy of "first optimize the architecture, then tune the parameters," the model's performance was significantly improved. Ultimately, through systematic hyperparameter tuning, the model's accuracy on the validation set reached a high of 84.52%, fully demonstrating the effectiveness of this optimization process.

4. Conclusion and future work

4.1. Summary of this paper

This study systematically investigated the application and optimization of BERT-based fine-tuned models in natural language processing tasks. Given that the "pre-training and fine-tuning" paradigm is powerful yet the fine-tuning process is highly sensitive to model architecture and hyperparameter settings, this paper proposed and implemented a structured two-stage optimization process, from "architecture exploration" to "parameter optimization," aiming to provide a set of effective practical solutions for improving model performance on specific downstream tasks.

Taking the MRPC sentence-pair classification task from the GLUE benchmark as the experimental platform, the main work and core conclusions of this study can be summarized as follows:

1. Validated the "Simpler is Better" Fine-tuning Strategy: In the first stage of fine-tuning strategy exploration, this study found that for the MRPC task, the features extracted by the BERT encoder are already highly separable. Therefore, a simpler single-layer linear classifier (test set accuracy of

DOI: 10.54254/2755-2721/2025.LD27825

83.30%) outperformed more complex structures that include multiple network layers and Dropout. This finding challenges the intuitive notion that "deeper models are better" and confirms that appropriately simplifying the model's top-level architecture is an effective optimization path for specific tasks.

2. Revealed the Nonlinear Impact of Key Hyperparameters: After determining the optimal classifier architecture, the second stage of large-scale hyperparameter tuning experiments intuitively and quantitatively revealed the profound impact of each parameter on model performance. The experiments showed that learning rate, weight decay, and warmup ratio all have an "optimal range," and deviating from this range leads to a performance drop. Among them, batch size was confirmed to be the most significant hyperparameter, and its improper selection (such as 64 in this experiment) can even cause a "cliff-like" drop in model performance.

3. Demonstrated the Great Value of a Systematic Optimization Process: Through the two-stage optimization process proposed in this study, the model's performance was significantly and stably improved. The experimental results show that through refined architecture selection and parameter tuning, the model's validation set accuracy can be increased from an unoptimized ~79% to a high of 84.52%. This fully proves that a systematic and rigorous fine-tuning optimization process, rather than relying on default settings or random trial-and-error, is a necessary condition for fully unleashing the potential of pre-trained models. As shown in the research by Sujatha and Nimala, combining multiple pre-trained models and conducting fine-grained parameter tuning can further enhance the performance of classification tasks [15].

In summary, this paper not only provides specific optimization parameter references for fine-tuning BERT on sentence-pair classification tasks but, more importantly, offers a transferable, structured optimization methodology. It emphasizes that in practice, attention should be paid to both the adaptability of the model architecture and the refined tuning of the training process.

4.2. Future work

Although this study has reached some valuable conclusions, there are still several limitations, which also point the way for future research:

Extension to More Tasks and Datasets: The conclusions of this study are primarily based on the MRPC, a medium-sized sentence-pair classification task. In the future, it is necessary to generalize and validate this optimization process on more types of tasks (such as sequence labeling, text generation), more domains, and datasets of different scales to test its universality. For example, when processing long documents that exceed the model's maximum length limit, special strategies such as chunking or hierarchical processing may be required [16].

Application to Different Pre-trained Models: This study focused on the bert-base-uncased model. With the emergence of more advanced and diverse pre-trained models like RoBERTa, ALBERT, and DeBERTa, future work could explore whether these different model architectures exhibit similar sensitivities to fine-tuning strategies and hyperparameters, or if there are model-specific optimization patterns.

Exploration of More Efficient Fine-tuning Paradigms: As the number of model parameters grows explosively, the computational cost of full fine-tuning the entire model is becoming increasingly prohibitive [14]. In the future, combining the optimization ideas of this study with Parameter-Efficient Fine-tuning (PEFT) techniques (such as LoRA [17], Adapter, Prompt Tuning, etc.) will be a very valuable direction. Investigating whether these efficient fine-tuning methods also follow the principle of "simplifying the classifier" and how sensitive they are to hyperparameters will be of great significance for promoting the application of large models in resource-constrained scenarios.

DOI: 10.54254/2755-2721/2025.LD27825

At the same time, methods like knowledge distillation can be explored to transfer the knowledge from large models to smaller, more efficient models, such as DistilBERT [18].

Adoption of More Advanced Optimization Algorithms: This study used Grid Search for hyperparameter tuning, which, while systematic, is computationally expensive. Future research could introduce more advanced automated machine learning (AutoML) techniques, such as Bayesian Optimization and genetic algorithms, to explore the vast hyperparameter space more efficiently and at a lower cost to find the global optimum [19]. Furthermore, one could draw inspiration from hybrid strategies like BlendSearch, which combines Bayesian optimization and local search as applied in the EcoOptiGen framework. Introducing cost-based pruning and progressive subsampling methods could also significantly improve optimization efficiency by eliminating ineffective configurations early in the evaluation of each hyperparameter combination, which is especially relevant for the increasingly costly fine-tuning of large models.

References

[1] Devlin, J., Chang, M.W., Lee, K. and Toutanova, K. (2019) BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. 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), 4171-4186.

[2] Peters, M.E., Ruder, S. and Smith, N.A. (2019) To Tune or Not to Tune? Adapting Pre-Trained Representations to Diverse Tasks. Proceedings of the 4th Workshop on Representation Learning for NLP, 7-14.

[3] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., ..., Polosukhin, I. (2017) Attention Is All You Need. Advances in Neural Information Processing Systems, 5998-6008.

[4] Treviso, M., Lee, J.U., Ji, T., van Aken, B., Cao, Q., ..., Schwartz, R. (2023) Efficient Methods for Natural Language Processing: A Survey. Transactions of the Association for Computational Linguistics, 11, 826-860.

[5] Phang, J., Fevry, T. and Bowman, S.R. (2018) Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-Data Tasks. arXiv preprint arXiv: 1811.01088.

[6] Hao, Y., Dong, L., Wei, F. and Xu, K. (2020) Investigating Learning Dynamics of BERT Fine-Tuning. Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing.

[7] Liu, X. and Wang, C. (2021) An Empirical Study on Hyperparameter Optimization for Fine-Tuning Pre-Trained Language Models. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2286-2300.

[8] Arase, Y. and Tsujii, J. (2021) Transfer Fine-Tuning of BERT with Phrasal Paraphrases. Computer Speech & Language, 66, 101164.

[9] Brickman, J., Gupta, M. and Oltmanns, J.R. (2025) Large Language Models for Psychological Assessment: A Comprehensive Overview. Advances in Methods and Practices in Psychological Science, 8, 1-26.

[10] Wang, C., Liu, S.X. and Awadallah, A.H. (2023) Cost-Effective Hyperparameter Optimization for Large Language Model Generation Inference. AutoML Conference.

[11] Sun, C., Qiu, X., Xu, Y. and Huang, X. (2019) How to Fine-Tune BERT for Text Classification? China National Conference on Chinese Computational Linguistics, 194-206.

[12] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O. and Bowman, S.R. (2018) GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. arXiv preprint arXiv: 1804.07461.

[13] Dolan, W.B. and Brockett, C. (2005) Automatically Constructing a Corpus of Sentential Paraphrases. Proceedings of the Third International Workshop on Paraphrasing.

[14] Dodge, J., Ilharco, G., Schwartz, R., Farhadi, A., Hajishirzi, H. and Smith, N. (2020) Fine-Tuning Pretrained Language Models: Weight Initializations, Data Orders, and Early Stopping. arXiv preprint arXiv: 2002.06305.

[15] Sujatha, R. and Nimala, K. (2024) Classification of Conversational Sentences Using an Ensemble Pre-Trained Language Model with the Fine-Tuned Parameter. Computers, Materials & Continua, 78, 1669-1686.

[16] Kong, J., Wang, J. and Zhang, X. (2022) Hierarchical BERT with an Adaptive Fine-Tuning Strategy for Document Classification. Knowledge-Based Systems, 238, 107872.

[17] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ..., Chen, W. (2022) LoRA: Low-Rank Adaptation of Large Language Models. International Conference on Learning Representations.

DOI: 10.54254/2755-2721/2025.LD27825

[18] Sanh, V., Debut, L., Chaumond, J. and Wolf, T. (2019) DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter. NeurIPS EMC2 Workshop.
[19] Schwartz, R., Dodge, J., Smith, N.A. and Etzioni, O. (2020) Green AI. Communications of the ACM, 63, 54-63.