Enhancing MobileBERT with Dynamic Data Augmentation and LoRA Fine-Tuning

Gil Biton
gilbito@post.bgu.ac.il

Roye Braverman
royebra@post.bgu.ac.il

March 25, 2025

Abstract

MobileBERT is a compact variant of the BERT architecture designed for efficient deployment on resource-constrained devices. Despite its relatively small size and fast inference, there remain opportunities to further improve its robustness, especially for tasks involving diverse or noisy language inputs. In this work, we propose enhancing MobileBERT with *dynamic data augmentation* strategies and a parameter-efficient *LoRA* (Low-Rank Adaptation) fine-tuning approach. We apply real-time text transformations such as synonym replacement, word swapping, and noise insertion during training to capture wider linguistic variations, and we adapt only low-rank parameters via LoRA to reduce memory overhead. Experimental comparisons against standard MobileBERT fine-tuning will evaluate our method's effectiveness in terms of accuracy, robustness, and efficiency on benchmark datasets. Our goal is to demonstrate that dynamic data augmentation and LoRA can synergistically improve MobileBERT's generalization on real-world textual inputs while retaining its core advantage of lightweight deployment.

1 Introduction

BERT's transformer-based architecture provides state-of-the-art results across tasks like sentiment analysis, question answering, and text classification, largely due to extensive pre-training on large corpora. However, its massive parameter counts and memory demands are prohibitive for mobile or embedded deployments.

MobileBERT addresses these constraints through a distillation-based training process and inverted-bottleneck layers, preserving much of BERT's representational power while reducing size and latency. Nevertheless, performance can still degrade on out-of-domain text (e.g., social media or specialized documents), and fully fine-tuning MobileBERT for each new task remains resource-intensive.

We tackle these issues with two enhancements. First, *dynamic data augmentation* injects varied textual perturbations (e.g., synonyms, swapped words, minor noise) at training time, boosting robustness to noisy or user-generated text. Second, *LoRA* (Low-Rank Adaptation) fine-tuning updates only small decomposed weight matrices, reducing memory overhead compared to retraining all parameters. We hypothesize that combining on-the-fly augmentation with LoRA will enable MobileBERT to handle diverse inputs while retaining its compact footprint. The following sections review relevant concepts, detail our approach, and describe our experimental evaluation.

2 Background

2.1 From BERT to MobileBERT

Pre-trained language models have significantly advanced NLP across tasks such as text classification, question answering, and sentiment analysis [1]. However, the original BERT models are computationally expensive and memory-intensive, making them less suitable for on-device or low-latency scenarios (e.g., mobile applications, embedded systems).

MobileBERT [2] addresses these limitations by compressing the BERT architecture while retaining sufficient representational capacity for downstream tasks. As shown in Figure 1, its design hinges on three core principles:

- **Bottleneck blocks**: Each Transformer layer uses a narrowed hidden dimension (e.g., 128), reducing parameter count while preserving deeper contextual modeling.
- **Teacher–Student Distillation**: A larger *IB-BERT* (Inverted-Bottleneck BERT) teacher model provides layer-wise guidance, enabling MobileBERT to inherit much of BERT's rich linguistic knowledge without requiring a full-scale parameter budget.
- **Stacked FFN Modules**: Additional feed-forward sub-layers compensate for the narrower hidden dimension, helping MobileBERT maintain non-linear capacity comparable to larger models.

By combining these strategies, MobileBERT sustains competitive accuracy while significantly lowering latency and memory usage compared to standard BERT.

2.2 Model Configurations

MobileBERT can be viewed as a "thin" variant of BERT$_{\text{LARGE}}$ tailored specifically to resource-constrained environments. Figure 2 below compares key configuration details among BERT$_{\text{LARGE}}$,

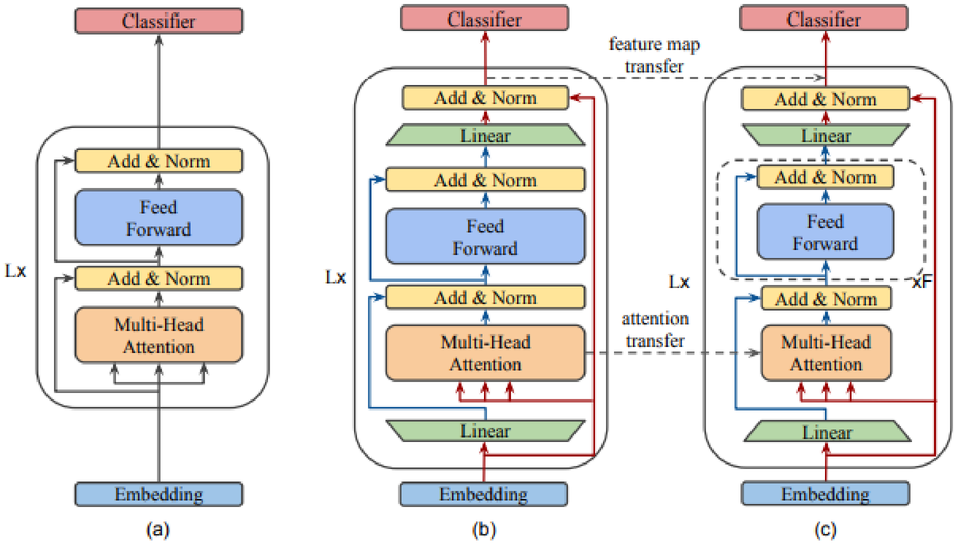
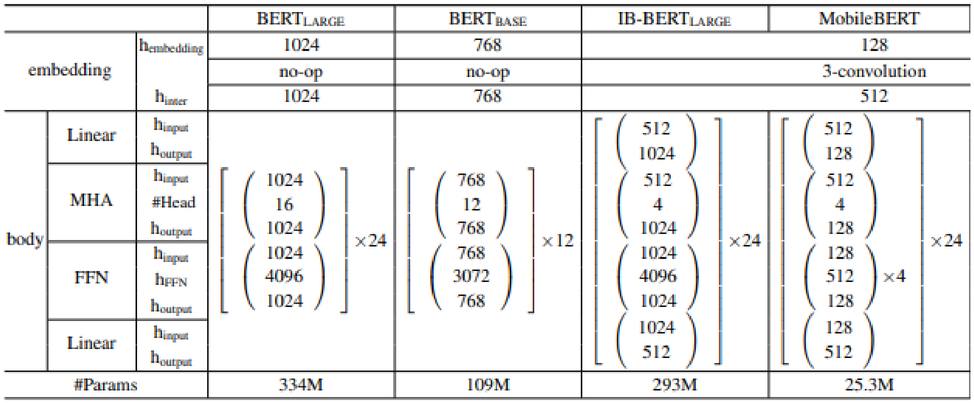


Figure 1: *Comparison of three architectures: (a) BERT; (b) Inverted-Bottleneck BERT (IB-BERT); and (c) MobileBERT. MobileBERT is trained via layer-to-layer imitation of IB-BERT, aligning both feature maps and attention distributions [2].*

BERT$_{\text{BASE}}$, IB-BERT$_{\text{LARGE}}$, and MobileBERT, including hidden sizes, attention heads, and feed-forward expansions. Typically, MobileBERT comprises around 25 million parameters, which is far fewer than the 110 million parameters of BERT$_{\text{BASE}}$.



			BERT$_{\text{LARGE}}$	BERT$_{\text{BASE}}$	IB-BERT$_{\text{LARGE}}$	MobileBERT
embedding		$h_{\text{embedding}}$	1024	768	128	
			no-op	no-op	3-convolution	
		h_{inter}	1024	768	512	
body	Linear	h_{input}			512	512
		h_{output}			1024	128
	MHA	h_{input}	1024	768	512	512
		#Head	16	12	4	4
		h_{output}	1024	768	1024	128
	FFN	h_{input}	1024	768	1024	128
		h_{FFN}	4096	3072	4096	512 ×4
		h_{output}	1024	768	1024	128
	Linear	h_{input}			1024	128
		h_{output}			512	512
			×24	×12	×24	×24
#Params			334M	109M	293M	25.3M

Figure 2: *An excerpt from the original MobileBERT architecture. It highlights differences in hidden sizes, feed-forward expansions, and the number of multi-head attention blocks [2].*

In practice, the MobileBERT training pipeline typically involves:

- **Inverted-Bottleneck Teacher (IB-BERT$_{\text{LARGE}}$)**: A larger teacher model is pre-trained on extensive text corpora (e.g., using masked language modeling), establishing a rich linguistic foundation.
- **Layer-wise Distillation**: MobileBERT is trained to mirror the teacher's intermediate feature

maps and attention distributions, allowing it to acquire robust language understanding within a compact architecture.

- **Fine-Tuning**: Once distilled, MobileBERT can be quickly adapted for specialized tasks (e.g., sentiment analysis, natural language inference) with substantially fewer resources than would be required for a full-sized BERT.

By integrating careful architectural compression with teacher–student distillation, MobileBERT achieves a balance between efficiency and contextual representation. Its streamlined footprint makes it especially attractive for scenarios where speed, memory, or battery life are at a premium, yet robust language understanding remains essential.

3 Methodology

3.1 Dynamic Data Augmentation

Real-world text often includes typos, unusual phrasing, or domain-specific language absent from standard corpora. We implement an **on-the-fly augmentation** strategy, where in each epoch, a random subset of training examples is modified using one of:

1. **Synonym Replacement:** Tokens are replaced with synonyms from WordNet, adding lexical variety.

2. **Random Word Swap:** Randomly permutes positions of some words to challenge strict order dependencies.

3. **Typo Injection:** Introduces small character-level errors to simulate user keyboard mistakes.

By applying these methods stochastically at each epoch, we generate a broader range of training inputs, potentially boosting MobileBERT's robustness against input noise.

3.2 LoRA: Parameter-Efficient Fine-Tuning

Adapting all of MobileBERT's parameters for each new domain or task can still be relatively expensive. We employ **LoRA** (Low-Rank Adaptation) [3] to freeze the majority of the model and only learn small rank-r matrices $\{A, B\}$:

$$W_{\text{adapted}} = W_{\text{frozen}} + \Delta W, \tag{1}$$

$$\Delta W = A \times B, \quad A \in \mathbb{R}^{d \times r}, \quad B \in \mathbb{R}^{r \times d}. \tag{2}$$

This reduces memory usage and accelerates fine-tuning while retaining MobileBERT's base knowledge from pre-training.

This approach significantly reduces the number of trainable parameters while retaining MobileBERT's base knowledge from pre-training. For example, in our experiments we observed:

As shown in Table 1, LoRA fine-tuning only requires updating $\sim 0.70\%$ of the total parameters, which lowers both memory usage and computational overhead. This allows rapid experimentation and deployment of specialized models on resource-constrained devices without retraining or storing a fully separate set of large weights.

Model	Trainable Params
MobileBERT	24,755,972
MobileBERT + LoRA	**173,058**

Table 1: Comparison of trainable parameters for MobileBERT vs. MobileBERT + LoRA.

4 Experiments & Results

Below, we outline our experimental setting with placeholders for the final results. We will specifically test **BoolQ** (a reading comprehension / QA-style classification task) rather than SQuAD, aligning with the model's ability to handle more natural queries.

4.1 Datasets

- **GLUE**: A diverse benchmark that consolidates multiple natural language understanding (NLU) datasets, each designed to test different aspects of linguistic reasoning and comprehension. Specifically, GLUE includes:
  - **CoLA**: Determines whether a given sentence is grammatically acceptable in English.
  - **SST-2**: Predicts the sentiment (positive/negative) of a movie review snippet.
  - **MRPC**: Classifies if two sentences express the same meaning (are paraphrases).
  - **STS-B**: Estimates how similar two sentences are on a continuous scale from 1 to 5.
  - **QQP**: Detects whether two questions from Quora share the same underlying intent.
  - **MNLI**: Judges if the relationship between a premise and hypothesis is entailment, contradiction, or neutral across multiple text genres.
  - **QNLI** : Determines whether a provided context sentence contains the answer to a posed question.
  - **RTE**: Predicts if one sentence logically entails another, aggregated from various textual entailment challenges.
  - **QNLI**: Resolves complex coreference ambiguities (though typically excluded due to known issues in evaluation).

  The combined tasks in GLUE cover sentiment analysis, paraphrase detection, textual entailment, and beyond, making it a comprehensive measure of a model's capability to understand and reason about language.

- **BoolQ**: A yes/no QA dataset where the model decides if a passage answers a particular question [4].

4.2 Training Details

To isolate the effect of data augmentation and eliminate tuning bias, we adopt the same training hyperparameters as defined in the original MobileBERT paper [2]. Specifically:

- **Batch size**: 16, 32, or 48, depending on dataset size and memory constraints.

- **Learning rate**: Between 1×10^{-5} and 1×10^{-4}.

- **Epochs**: Between 2 and 10, selected based on development set performance.
- **Augmented fraction**: 10% of training samples are dynamically augmented in each epoch to improve generalization without destabilizing training.

4.3 Effect of Different Augmentation Ratios

We further investigated how different augmentation ratios (10%, 20%, 30%, 40%) influence the model's performance during training. Figures 3 and 4 show evaluation metrics (MCC for CoLA and F1 for QNLI) as a function of training steps:

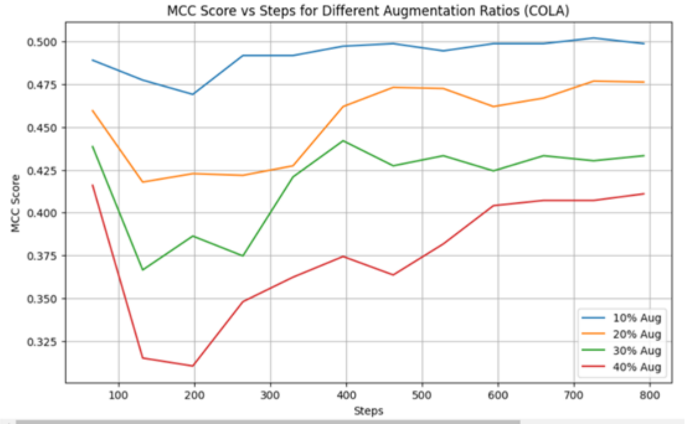


Figure 3: MCC Score vs. Steps for varying augmentation ratios on CoLA. Higher is better.

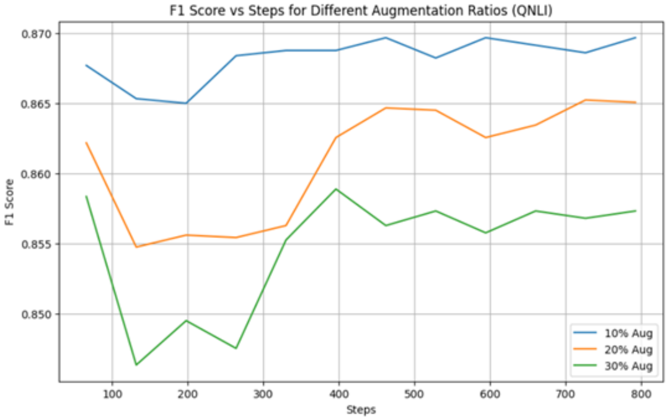


Figure 4: F1 Score vs. Steps for varying augmentation ratios on QNLI.

These experiments suggest that *light to moderate* augmentation percentages (e.g., 10–20%) often achieve the best balance between diversity and stability, especially for tasks sensitive to syntactic or semantic alterations. Stronger augmentation can still catch up later, but may require more careful hyperparameter tuning to avoid early performance dips. experiments suggest that *light to moderate* augmentation percentages (e.g. 10–20%) often achieve the best balance between diversity and stability, especially for tasks sensitive to syntactic or semantic alterations. Stronger augmentation can still catch up later, but may require more careful hyperparameter tuning to avoid early performance dips.

4.4 Experimental Evaluation

	MNLI	RTE	QQP	QNLI	MRPC	CoLA	SST-2	STSB	BoolQ
Augmented MobileBERT	82.05	**66.78**	**89.09**	**90.70**	83.57	**70.98**	90.57	**87.65**	**73.21**
MobileBERT	**83.30**	66.20	70.20	90.60	**88.80**	50.50	**92.80**	84.40	62.17

Table 2: Results of augmented MobileBERT vs regular mobilebert on GLUE BoolQ tasks.

Overall, these results indicate that dynamic augmentation, even at modest levels, can yield clear benefits in domains where lexical or syntactic variation is critical. However, striking the right balance between original and augmented data remains an important consideration for maximizing performance across a diverse range of NLP tasks.

5 Conclusion and Future Directions

In this work, we enhanced MobileBERT through dynamic data augmentation and LoRA-based fine-tuning, aiming for greater robustness in noisy or domain-shifted settings and more efficient adaptation to new tasks. Our approach remains faithful to MobileBERT's core advantage—its compactness—while offering improved flexibility for real-world deployment.

Looking forward, several directions arise from our project presentation:

- **Domain-Specific Applications:** Extending the augmented MobileBERT model to specialized areas such as legal or biomedical text.
- **Advanced Augmentation:** Incorporating back-translation, context-aware replacements, or other sophisticated methods to further boost generalization.
- **Integration with Edge AI:** Deploying the lightweight model on edge devices for real-time NLP, possibly combined with quantization or pruning for extreme resource constraints.

Link for github : https://github.com/gilbiton1/gilbiton1

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[2] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou. MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices. *arXiv preprint arXiv:2004.02984*, 2020.

[3] E. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, L. Wang, and W. Chen. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*, 2021.

[4] C. Clark, M. Yatskar, and L. Zettlemoyer. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. *NAACL*, 2019.

6 Appendix: Additional Plots

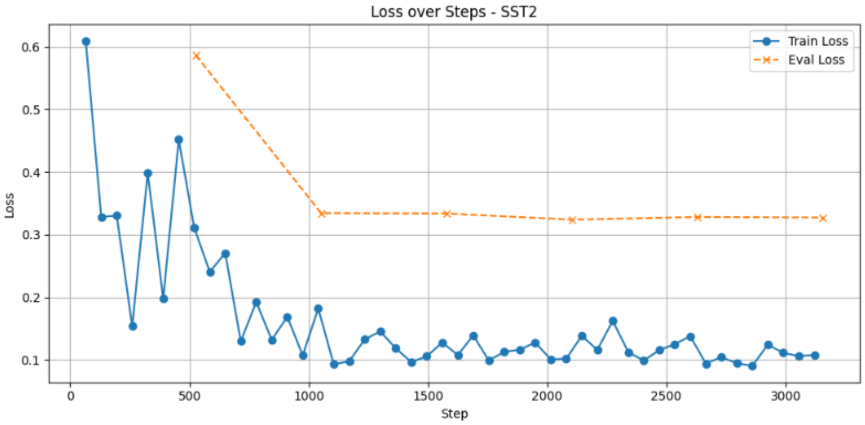


Figure 5: Training and Validation Loss curves over augmented steps for the SST-2 dataset.

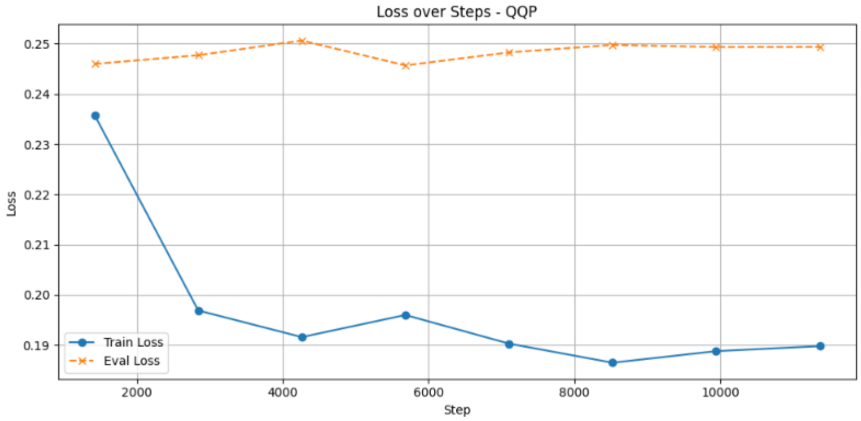


Figure 6: Training and Validation Loss curves over steps for the QQP dataset.