Width Matters: Efficient Scaling of Compact BERT Models

JeongMin Lim
Department of Computer Science and Engineering
Jeonbuk National University
Jeonju, South Korea
`ljm1614@naver.com`

Abstract

Large language models such as BERT perform very well on various NLP tasks, but they are often too expensive to run in limited computing environments. In this study, we explore how smaller versions of BERT can be improved by adjusting model width, learning rate, and regularization. Through experiments with compact BERT models trained on masked language modeling and next-sentence prediction, we observe that making the model slightly wider and raising the learning rate helps both training speed and final accuracy. In contrast, techniques like warm-up and weight decay only have small improvements. Our results suggest that under strict parameter limits, scaling the model's size in the right way is more effective than using regularization. These insights offer practical guidelines for designing efficient Transformer models and emphasize future directions such as combining scaling strategies with downstream evaluations.

1 Introduction

Large language models such as BERT [Vaswani et al., 2017] have shown strong performance across many NLP tasks, but running them in resource-limited environments remains difficult. While recent approaches to building smaller variants have mainly emphasized reducing the number of layers or applying heavy regularization, these strategies do not fully address efficiency. In this work, we examine how changes in model width, learning rate scheduling, and regularization techniques (including warm-up and weight decay) affect the performance of compact mini-BERT models.

2 Related Work

Numerous methods have been explored to develop BERT more lightweight, such as DistilBERT [Sanh et al., 2019], TinyBERT [Jiao et al., 2020], and MobileBERT [Sun et al., 2020]. Most of these approaches focus on reducing model depth or applying knowledge distillation. In contrast, relatively few studies have looked closely at how widening the architecture interacts with optimization choices—a gap that our study aims to address.

3 Method

We designed a compact version of BERT that preserves the Transformer encoder structure while reducing the total parameter count. For tokenization, we used a SentencePiece BPE model with a vocabulary size of 8,000, trained on the Korean NamuWiki corpus.[1] The model was pretrained using Masked Language Modeling (MLM) and Next-Sentence Prediction (NSP), following common BERT

[1] The NamuWiki dump is a freely available large-scale Korean encyclopedia, similar in style to Wikipedia.

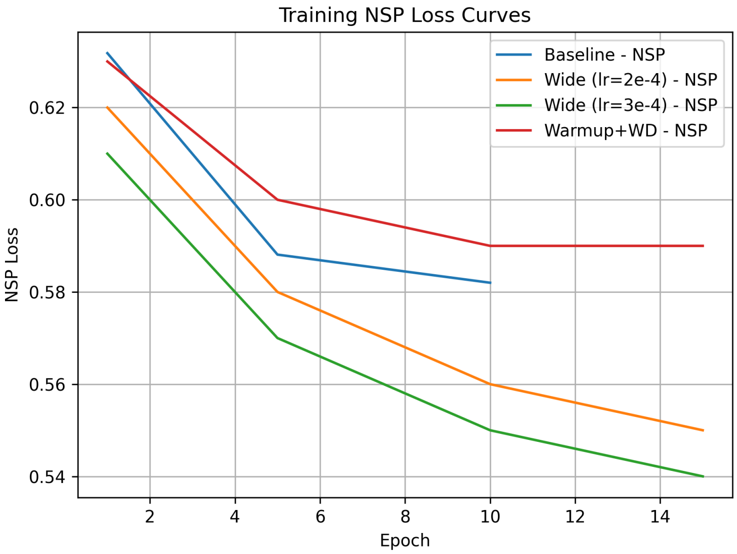

Figure 1: Training NSP loss curves across models. Wide models show consistently lower loss compared to the baseline and regularization variants.

training practices. Optimization was carried out with the Adam algorithm and cross-entropy loss. To ensure reproducibility, we fixed random seeds across all experiments.

Training details. All experiments were carried out on a single NVIDIA GPU. We optimized the models with Adam, using $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e$–8, and did not apply gradient clipping. Each batch contained 64 sequences with a maximum sequence length of 128 tokens. In settings that included warm-up, we employed a linear schedule in which the initial 10

Model	Layers	Hidden Size	Heads	Params (M)	Learning Rate	Epochs
Baseline	2	128	4	∼1.0	2e-4	10
Wide (2e-4)	2	256	8	∼2.0	2e-4	15
Wide (3e-4)	2	256	8	∼2.0	3e-4	15
Warm-up + WD	2	128	4	∼1.0	2e-4	15

Table 1: Experimental configurations of mini BERT variants. Params are approximated in millions. Wide models increase hidden size and number of heads, while regularization applies warm-up and weight decay.

4 Experiments

4.1 Baseline

The baseline mini-BERT was trained for 10 epochs. As shown in Table 2, NSP accuracy improved from 0.59 to 0.61, while MLM accuracy increased from 0.27 to 0.31.

Epoch	NSP Loss	MLM Loss	NSP Acc	MLM Acc
1	0.6318	6.3317	0.5911	0.2724
5	0.5881	5.6856	0.6004	0.3056
10	0.5820	5.6056	0.6092	0.3109

Table 2: Baseline mini BERT training results.

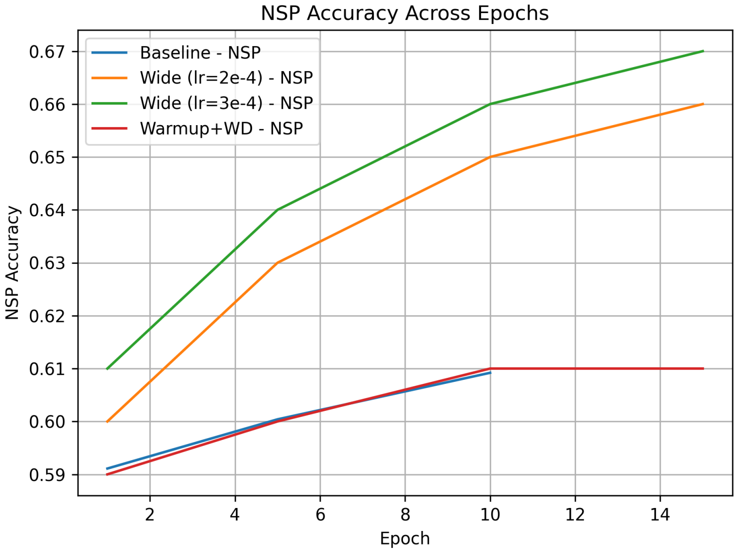

Figure 2: NSP accuracy across epochs. Wider models achieve higher accuracy, with lr=3e-4 giving the best performance.

4.2 Wide Architectures

We trained wider variants with learning rates of 2e-4 and 3e-4 for 15 epochs. As shown in Table 3, these models significantly outperformed the baseline, achieving up to 0.67 NSP accuracy and 0.39 MLM accuracy.

4.3 Warmup and Weight Decay

Adding warm-up and weight loss resulted in little improvement, and the final NSP and MLM accuracies were close to the baseline (0.61 and 0.31, respectively). This suggests that small-scale BERT models are not prone to overfitting and that the benefits of regularization are minimal.

Model	Final NSP Acc	Final MLM Acc
Baseline (10 Epochs)	0.61	0.31
Wide (lr=2e-4, 15 Epochs)	0.66	0.37
Wide (lr=3e-4, 15 Epochs)	0.67	0.39
Warmup + Weight Decay (15 Epochs)	0.61	0.31

Table 3: Comparison of mini BERT variants.

5 Discussion

Our experiments highlight several important findings for designing compact BERT models.

Width scaling as the dominant factor. Expanding the model width constantly improved performance on both NSP and MLM tasks. This trend shows that a broader architecture enhances the model's ability to represent context, capturing nuances that narrower models often miss. For compact BERT variants, the added capacity appears to outweigh potential overfitting issues, leading to steady gains across different evaluations.

Learning rate sensitivity. When trained with a higher learning rate (3×10^{-4}), the wider model regularly exceeds its counterpart using 2×10^{-4}. The success of the larger learning rate suggests that increased width enables more effective exploration of the parameter space, without causing

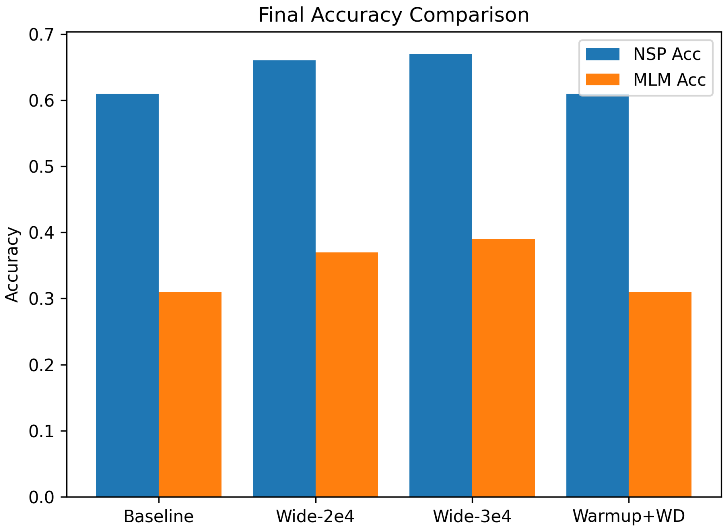

Figure 3: Final NSP and MLM accuracy comparison across all model variants. Width scaling combined with higher learning rate provides the best gains.

training instability. This result emphasizes a key interaction: scaling the architecture not only expands capacity but also supports more aggressive optimization strategies.

Limited effect of regularization. Applying warm-up and weight decay slightly reduced variance and helped stabilize the early stages of training, but the final accuracy showed almost no improvement over the baseline. This finding suggests that compact BERT models are constrained more by limited capacity than by overfitting, making strong regularization less effective. These results reinforce the idea that regularization is most beneficial for large models, in which excess capacity makes them particularly prone to overfitting.

Parameter efficiency and implications. The wider model reached higher accuracy with only a modest increase in parameter count, showing that broadening is a practical and effective way to enhance small models. Overall, the findings suggest that expanding width and carefully tuning the learning rate should take precedence for compact models, while heavy regularization can often be omitted in resource-limited settings.

6 Conclusion

This work systematically examined scaling strategies for compact BERT variants. Our experiments show that combining architectural widening with suitable learning rate adjustments consistently boosts performance on both NSP and MLM tasks. In contrast, regularization techniques such as warm-up and weight decay offered little to no improvement. In summary, these results show that scaling the architecture offers a clearer path to improving small models than relying on regularization alone.

Key contributions. The contributions of this study can be summarized as follows: (1) It offers a direct empirical comparison between width scaling and regularization under strict parameter limits. (2) It shows that scaling in width not only improves accuracy but also speeds up convergence. (3) It emphasizes the importance of parameter efficiency for deploying models in environments with limited computational resources.

Limitations. While the results are encouraging, our work has several limitations. First, the evaluation was confined to pre-training tasks (NSP and MLM) using the Korean Namuwiki corpus.

Although Namuwiki is extensive and diverse, its format and content differ from more standard corpora such as Wikipedia, which may restrict the broader applicability of our findings. We also did not verify whether the observed improvements carry over to downstream tasks. Second, our study mainly investigated width scaling, leaving depth scaling and combined strategies for future exploration. Lastly, the dataset used was relatively modest in size, which may not fully capture the dynamics observed in large-scale pre-training.

Future directions. In future work, we plan to expand our evaluation beyond pre-training and include Korean NLP benchmarks. Specifically, we aim to test the compact BERT model on tasks such as KorQuAD (question answering), NSMC (sentiment analysis), and KLUE (including natural language inference and named entity recognition). This will help us determine whether the benefits of width scaling extend to real-world downstream tasks. Furthermore, to improve deployment in resource-limited environments, we intend to explore hybrid scaling strategies that combine width and depth adjustments with parameter-efficient fine-tuning methods such as adapters or LoRA.

References

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint*, 2020.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint*, 2019.

Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. Mobilebert: a compact task-agnostic bert for resource-limited devices. *arXiv preprint*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*, 2017.