Journal of Machine Learning Research 23 (2022) 1-11 Submitted 1/21; Revised 5/22; Published 9/22

Model Merging: Enhancing NLP Task Performance by Integrating Fine-Tuned Small Models with BERT Base

Iris Dania Jimenez IRIS.JIMENEZ@CAMPUS.LMU.DE
Department of Mathematics, Informatics and Statistics
Ludwig-Maximilians-Universität München

Editor: None

Abstract

This study explores the potential of combining lightweight NLP models like TinyBERT with large pre-trained models like BERT Base to enhance performance while reducing computational costs. Transfer learning, the act of fine-tuning foundation models on a specific task, delivers strong results but demands significant resources, making it impractical for many real-world applications. By evaluating the performance of a merged model on two datasets (IMDb and SNLI) in terms of accuracy and computational efficiency (FLOPs), this project aims to determine if a hybrid approach can provide a better balance of performance and resource utilization than transfer learning alone. Code available here.

Keywords: Model merging, BERT, NLP

1 Introduction

In the rapidly evolving field of Natural Language Processing (NLP), the quest for building more efficient and effective models is continuous. Traditional approaches often involve fine-tuning pre-trained models, like BERT Base (Devlin et al. (2019)), to achieve state-of-the-art performance on various NLP tasks. However, the computational costs and resource demands associated with fine-tuning these big models can be prohibitive, especially in real-world applications where efficiency is crucial.

Recent advancements have introduced smaller, more efficient models like TinyBERT (Jiao et al. (2020)), which are designed to retain much of the performance of their larger counterparts while significantly reducing computational requirements. These models have opened up new possibilities for creating lightweight NLP solutions. Yet, the question remains: can the strengths of these smaller models be effectively combined with the power of large pre-trained models to further enhance performance?

This study seeks to explore this possibility by evaluating whether merging a fine-tuned small model with a large pre-trained model can improve performance on specific NLP tasks, compared to the traditional approach of fine-tuning the large model alone. By leveraging, for fine-tuning, the GLUE benchmark (Wang et al. (2019)), which is widely recognized for benchmarking various NLP tasks, this study aims to provide insights into the potential benefits and trade-offs of model merging in practical applications. The primary focus will be on determining whether a hybrid approach can offer a superior balance of accuracy and resource utilization compared to transfer learning. To do this we will use BERT Base as the foundation model and fine-tuned TinyBERT as the lightweight model and we will compare their performance on two different datasets: IMDb and SNLI in terms of accurancy and FLOPs.

The structure of this study is as follows: Section 2 offers a comprehensive overview of model merging, discussing existing research in the field. Section 3 introduces the datasets used in the analysis, while Section 4 outlines the methodology and steps undertaken in the project. Section 5 presents the results, followed by conclusions in Section 6. Finally, Section 7 addresses the limitations of the study and suggests future research directions.

2 Model Merging

Model merging refers to a set of techniques designed to combine the parameters of different models into a single hybrid model, allowing the strengths of multiple models to be leveraged simultaneously. In the current AI landscape, powerful foundation models such as GPT-4 (Achiam et al. (2023)), BERT (Devlin et al. (2019)), Gemini (Anil et al. (2023)), and LoRA (Hu et al. (2021)), among others, require significant time and computational resources for both training and fine-tuning. While these models achieve remarkable performance across a variety of tasks, fine-tuning them for specific use cases can be prohibitively expensive. Model merging has emerged as a promising approach to address this issue, offering a more computationally efficient way to combine the vast knowledge stored in large pre-trained models with the task-specific adaptability of smaller, fine-tuned models. By merging the parameters of these models, we can create a hybrid model that retains the strengths of both, enabling enhanced performance without the need for extensive re-training or fine-tuning.

At first glance, model merging may seem similar to ensemble learning, where predictions from multiple sources are combined to achieve superior results. However, merging operates at a deeper level, integrating the models themselves rather than just their predictions. This approach creates a unified new model that combines the knowledge from the original models at the parameter level, allowing it to efficiently exploit the strengths of multiple sources while minimizing the computational overhead typically associated with fine-tuning or inference. Figure 1 illustrates the distinction between model merging and ensemble learning, showcasing how merging creates a single model, while ensembling aggregates predictions from several models.

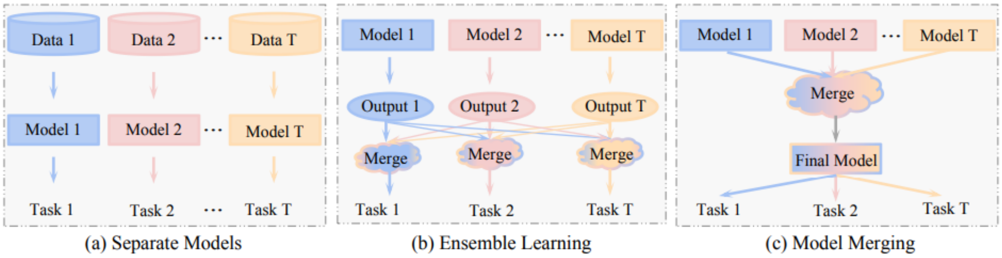

Figure 1: Model merging vs Ensemble learning (Yang et al. (2024a))

Model merging methods can be broadly divided into two categories: pre-merging and during-merging approaches (Yang et al. (2024a)). Pre-merging techniques are designed to align and prepare models before merging, ensuring that their parameters are compatible and that the integration process is smooth. One such method is linearization fine-tuning (Ortiz-Jimenez et al. (2023)), which disentangles the weight space from the input space. This disentanglement helps reduce interference between models during the merging process, ultimately enhancing the performance of the hybrid model. Another pre-merging technique is architecture transformation, which is used when models with different architectures need to be merged. For example, in (Avrahami et al. (2022)), the authors successfully merged models with different GAN architectures, while in (Nguyen et al. (2023)), the authors aligned models with different numbers of layers. Finally, weight alignment techniques, based on the linear mode connectivity property, aim to align the weights of models that share functional similarities, even if they were trained with different hyperparameters. Techniques such as CCAMerge (Horoi et al. (2024)) use this property to maximize the correlation between neurons, ensuring that the models are aligned for more accurate and effective merging. During-merging techniques, on the other hand, focus on the actual process of merging models by dynamically adjusting parameters during the merge. One common approach is weighted-based merging, where multiple models are combined by assigning optimal weights or coefficients to each model's parameters based on their relative importance. For instance, Evolutionary-Model-Merge (Akiba et al. (2024)) uses evolutionary algorithms to find the best merging coefficients, while AdaMerging (Yang et al. (2024b)) leverages gradient descent to minimize the entropy in the merged model's predictions, learning optimal coefficients as part of the process.

Another prominent category of during-merging methods is subspace-based merging, where models are projected into sparse subspaces to facilitate the merging process. The rationale behind this is inspired by model pruning, which has shown that removing a large portion of a model's parameters does not significantly affect its performance (Yadav et al. (2023)). The TIES-Merging technique (Yadav et al. (2023)) focuses on retaining only the top 20% of parameters with the largest magnitudes from each model, effectively merging them in a sparse subspace without losing critical information.

Finally, routing-based merging introduces a more dynamic and adaptive approach by adjusting the model's structure or parameters during inference based on the specific input or task. This method allows the merged model to be flexible and adaptable, optimizing its internal configuration in real-time depending on the task at hand. For example, SMEAR (Muqeeth et al. (2024)) calculates a weighted average of parameters from multiple expert models based on input, allowing the merged model to adjust its processing strategy dynamically. This enables a highly efficient, task-specific optimization, while maintaining computational costs comparable to using a single expert model (Muqeeth et al. (2024)).

Pre-merging methods offer significant advantages in terms of model compatibility and stability. By aligning models before merging, they reduce the risk of performance degradation and ensure smoother integration. However, these methods can be complex to implement and may not provide the flexibility to adapt during inference. In contrast, during-merging methods offer a more dynamic approach, allowing the model to adjust based on the specific input during inference. This flexibility often leads to more efficient task-specific performance. The downside is that these methods introduce more complexity during the inference phase, as the model must dynamically adjust its parameters on the fly.

In conclusion, model merging presents a powerful and efficient solution for combining the strengths of different models, whether large foundation models or smaller, fine-tuned models. By merging at the parameter level, model merging offers a resource-efficient alternative to transfer learning or ensemble learning. Depending on the specific strategy, whether pre-merging or during-merging, the methods explored in this section adapt to the needs of various tasks, offering flexibility, stability, and adaptability across a wide range of applications. As deep learning models grow increasingly larger and more resource-intensive to train, model merging could become a key technique for optimizing both performance and computational efficiency in increasingly complex model architectures.

3 Data

This section introduces the four datasets used in this study: SST-2 and RTE for fine-tuning and IMBDb and SNLI for testing. Each dataset represents a distinct NLP task, characterized by different goals and structures.

The Stanford Sentiment Treebank (SST-2) dataset is designed for binary sentiment classification. The task requires determining whether a given sentence expresses a positive or negative sentiment. Each sample is drawn from movie reviews and is labeled as either positive (1) or negative (0). The main performance metric for SST-2 is accuracy, which evaluates how often the model correctly predicts the sentiment label.

The Recognizing Textual Entailment (RTE) dataset is used for a binary classification task, where the objective is to determine whether a hypothesis logically follows from a given premise. Each data point consists of two sentences: the premise, which states a fact or situation, and the hypothesis, which might be inferred from that premise. The task is to classify whether the hypothesis is entailed by the premise, meaning the premise logically supports the hypothesis, or if it is not entailed, meaning the premise contradicts or doesn't support the hypothesis. Similar to the SST-2 dataset, the primary evaluation metric for RTE is accuracy.

The Internet Movie Database (IMDb) dataset is a dataset commonly used for sentiment analysis. It consists of 50,000 movie reviews, equally split between positive and negative sentiment labels. One key feature of the IMDb dataset is that it contains real-world user-generated content, which makes it an suitable resource for testing models in environments with varying language styles, colloquialisms, and informal expressions. Similarly to the previous datasets the performance metric is accuracy.

The Stanford Natural Language Inference (SNLI) dataset is a large-scale corpus designed for the task of natural language inference (NLI), where the goal is to determine the relationship between a pair of sentences. Similarly to RTE, each pair consists of a premise and a hypothesis, and the model must classify their relationship as one of three possible categories: entailment, contradiction, or neutral. The SNLI dataset contains 570,000 sentence pairs, making it one of the largest datasets for NLI tasks. It is widely used in training and evaluating models for tasks that require an understanding of sentence semantics, such as entailment detection and reasoning.

In summary, these datasets provide a diverse range of NLP tasks that enable a comprehensive evaluation of model performance across different challenges. The variety in task structure makes these datasets ideal for testing the effectiveness of model merging in both semantic similarity and classification contexts. Table 1 presents an example from each dataset.

Dataset	Task Description	Example
RTE	Does sentence A entail sentence B?	A) In a restaurant a man is eating a sandwich B) A man is eating = **Entailed**
SST-2, IMDb	Is the movie review positive, negative, or neutral	A) The movie is funny and smart = **Positive**
SNLI	Does sentence A entail sentence B?	A) A smiling child is holding a toy B) A happy child in costume holds a toy = **Neutral**

Table 1: Examples of the RTE, SST-2, IMDb and SNLI datasets (McCormick (2019))

4 Methodology

This section presents the methodology used to evaluate the effectiveness of model merging techniques across several NLP tasks. The goal is to merge the parameters of foundation models and fine-tuned lightweight models to assess whether the merged models can achieve comparable or superior performance to individually fine-tuned foundation models, with reduced or comparable computational costs.

The experiments were conducted using four well-known NLP datasets: SST-2 and RTE for fine-tuning and IMDb and SNLI for testing. These datasets were selected because they represent a wide spectrum of NLP tasks, ranging from common tasks like sentiment analysis (SST-2 and IMDb) to more complex reasoning tasks like sentence entailment (RTE and SNLI). Each dataset was preprocessed by tokenizing the text and formatting it appropriately for model input. Performance was evaluated using accuracy for all datasets. Additionally, FLOPs were measured to assess the computational efficiency of the models, making the evaluation comprehensive both in terms of model performance and hardware resource utilization.

The models used in this study include the pre-trained BERT Base model and a fine-tuned TinyBERT model. The decision to use BERT Base as the foundation model is due to its established reputation as a robust and widely adopted model. Specifically, BERT Base was chosen because it has a hidden size of 768, which is compatible with the 6-layer version of TinyBERT, which also has a hidden size of 768. In contrast, the 4-layer TinyBERT has a hidden size of 312, and the model merging technique implemented in this work requires both models to have the same hidden size. This constraint also ruled out BERT Large as foundation model, as its hidden size is 1024. Another potential candidate as lightweight model was DistilBERT (Sanh et al. (2020)), which also has a hidden size of 768. However, when comparing it to the 6-layer TinyBERT, both having the same number of encoder layers (12) and a similar number of parameters ($\sim$ 66 million), TinyBERT consistently performed better (Jiao et al. (2020)). For these reasons, TinyBERT was chosen over DistilBERT as the lightweight model.

The primary goal was to compare the performance of the merged models (pre-trained BERT Base combined with fine-tuned TinyBERT) against a fine-tuned BERT Base model, so for the merging process, the pre-trained BERT Base model was used without any additional fine-tuning, while TinyBERT was fine-tuned on the previously mentioned datasets.

Fine-tuning of the TinyBERT and BERT Base on the SST-2 and RTE datasets was conducted with a learning rate of 2e-5, batch size equal to 16, weight decay set to 0,01 and it was run for 3 epochs.

Model merging was conducted using a weighted-based merging technique, where the weights of the two models were combined by optimizing merging coefficients (alpha values). Specifically, the merged weights W_{merged} were computed as follows:

$$W_{merged} = \alpha \cdot W_{TinyBERT} + (1 - \alpha) \cdot W_{BERTBase} \tag{1}$$

where $W_{TinyBERT}$ and $W_{BERTBase}$ represents the parameters of the fine-tuned TinyBERT model and the pre-trained BERT Base model, respectively, and α is the coefficient found via random search. This search was conducted over twenty trial, with values that range between 0 and 1. Values closer to 1 indicate that merged model is relying more on the lightweight model, while values closer to 0 means that predictions are more affected by BERT Base's weights. Twenty trials were chosen as a balance between computational efficiency and a sufficiently broad exploration of the alpha space.

All experiments were conducted on a DGX A100 Architecture, which consists of 8 nodes, each with 256 CPU cores, 1 TB of memory, and 8 NVIDIA A100 GPUs, each providing 40 GB of GPU memory. The models were implemented and trained using PyTorch.

Following the merging of the models, performance was analyzed using the aforementioned evaluation metrics, and the results were compared against the baseline fine-tuned BERT Base. This comparison enabled to assess the effectiveness of the merging strategy both in terms of model accuracy and computational efficiency.

5 Results

This section presents the results obtain by following the previous described steps. Table 2 reports the optimal alpha value found via random search, Table 3 reports the performance obtain by both models on the IMDb and SNLI dataset. Results for the FLOPs metric are reported in TERAFLOPs. The TinyBERT model fine-tuned on SST-2 was merged with the pre-trained BERT Base and evaluated on the IMDb dataset, while the TinyBERT model fine-tuned on RTE was similarly merged with the pre-trained BERT Base and tested on the SNLI dataset.

	TinyBERT SST-2	**TinyBERT RTE**
Alpha	0,031	0,60

Table 2: Optimal alpha values

Model	**Accuracy**		**TFLOPs**	
	IMDb	**SNLI**	**IMDb**	**SNLI**
Merged model	52,56%	35,28%	108,74	106,83
BERT Base fine-tuned	85,76%	40,30%	108,74	106,83

Table 3: Accuracy and TFLOPs of BERT Base vs merged model

The results are quite surprising. On the IMDb dataset BERT Base fine-tuned on the SST-2 dataset achieves a 85% of accuracy; a solid result demonstrating the robust performance of BERT Base and the model's ability to generalize effectively. On the contrary the performance achieved by the merged model is way worse with an overall accuracy of 52%, suggesting almost a random performance of the merged model. This could be because it combines two models with different levels of training; the fine-tuned BERT Base model is specifically optimized for sentiment analysis (as in SST-2 and IMDb), whereas the merged model may lose that optimization due to the introduction of pretrained, non-task-specific weights from the larger BERT base model. The merging of fine-tuned and pretrained weights likely disrupted the parameters that were finely tuned for the task, leading to degraded performance. This suggest that even though TinyBERT was fine-tuned on SST-2, its small contribution to the merged model (0,031) doesn't carry enough weight to improve performance significantly suggesting that the small weight assigned to TinyBERT might not be sufficient to help the model generalize well to a dataset like IMDb. On the SNLI dataset the fine-tuned BERT Base achieves a very poor performance with a 40,30% accuracy. This could be due to a multiple of reasons. First, dataset size plays a crucial role in this performance gap. RTE is relatively small, with around 2.5K training examples, whereas SNLI is much larger, containing about 550K training examples. As a result, a model fine-tuned on the smaller RTE dataset may lack the exposure to a broad variety of patterns necessary for robust generalization to larger datasets like SNLI. Additionally, the label distribution differs between the two tasks. RTE focuses on binary classification, distinguishing between entailment and non-entailment, whereas SNLI is a three-class classification task, requiring the model to differentiate between entailment, neutral, and contradiction. This additional complexity in SNLI likely causes the RTE-fine-tuned model to struggle with more nuanced distinctions. Furthermore, domain differences exacerbate the problem: RTE often consists of formal language from news or legal texts, while SNLI includes more conversational, everyday language, particularly from image captions. These differences in task structure and language domain contribute to the model's markedly inadequate performance on SNLI. The drop in performance between the fine-tuned BERT Base and merged model is only a 5%, with the merged model achieving an overall 35,28% accuracy, while on the IMDb dataset the difference in performance between BERT Base and the merged model was 33%. For the SNLI dataset, the two models were merged using an alpha value of 0.60, meaning that the merged model relied more heavily on the weights of the TinyBERT model fine-tuned on the RTE dataset. This higher reliance on TinyBERT could explain why the performance gap between the merged model and the fine-tuned BERT Base model was not as pronounced as it was in other tasks, such as the IMDb dataset. In this case, the model benefited more from the task-specific knowledge captured by the fine-tuned TinyBERT on a related natural language inference task (RTE), which likely enabled the merged model to better handle the nuances of SNLI, another natural language inference task. In terms of FLOPs there is no noteably difference between the two models, both for the IMDb and the SNLI dataset.

6 Conclusions

Overall, these results suggest that merging models with different levels of fine-tuning can lead to substantial degradation in performance, instead of a performance gain, particularly when the pre-trained model is not adequately fine-tuned for the target task. This may occur because pre-trained models, while capable of handling a wide variety of general language tasks, lack the specific optimizations necessary for achieving high performance in task-specific domains like sentiment analysis or natural language inference. Fine-tuning helps models specialize by adjusting their weights based on task-relevant features and patterns. The fact that we are using a large foundation model like BERT Base does not seems to guarantee generalization to a task unless it is fine-tuned for that task, even when merged with a lightweight fine-tuned model. The merging of fine-tuned and pre-trained models can dilute the task-specific knowledge captured by the fine-tuned model, instead of enhacing it by leveraging the knowledge of the foundation model. However, in cases where the fine-tuned model contributes more significantly, as indicated by a higher alpha value during the merging process, the merged model may still retain some of the task-specific knowledge from the fine-tuned model, leading to a smaller performance gap. In such cases, the task-specific knowledge from the fine-tuned model can still influence the final model's decision-making process, preserving some of the specialized patterns it has learned. However, when the task is slightly different from what the fine-tuned model was trained on, such as testing RTE fine-tuned TinyBERT and BERT Base on SNLI, both fine-tuned models do not seem to generalize well, and performance drops significantly. Importantly, there is no significant difference in FLOPs between the merged model and the fine-tuned BERT Base model, with both models requiring comparable computational resources for inference. This could indicate that the performance drop in the merged models is not due to a reduction in computational capacity but rather a consequence of the merging technique.

It's important to note that these conclusions are based on the weighted merging technique, where model parameters are interpolated between pre-trained and fine-tuned models based on a learned alpha value. While this technique aims to balance the general language understanding of the pre-trained model with the task-specific knowledge of the fine-tuned model, it may not be optimal in this case. The merging of models with different levels of fine-tuning might disrupt the specialized parameter distributions that are crucial for the task, particularly when the majority of the weights come from a pre-trained model that hasn't been tailored for the specific task. This highlights an important fact: the success of model merging might depend heavily on the proportion of task-specific knowledge retained from fine-tuning rather than the knowledge from the foundation model, and the merging technique itself must be carefully planned. Therefore, careful consideration must be given to the merging process, particularly the balance between pre-trained and fine-tuned weights, to ensure that the final model is capable of achieving optimal performance for the target task. Despite the similar computational costs, the performance gap suggests that the merging strategy may not fully leverage the strengths of both models, and alternative techniques or a greater reliance on the fine-tuned model may be needed to avoid significant performance losses, especially in tasks that require domain-specific expertise.

7 Limitations and Future Directions

The results of this study highlight several limitations that warrant further investigation. First, the weighted merging technique used to combine the pre-trained BERT base model with fine-tuned TinyBERT was not optimal. This suggests that the current approach to merging models may not fully exploit the strengths of both models, leading to suboptimal results. Future directions could include experimenting with alternative model-merging methods, such as layer-wise merging or adaptive merging, which might better preserve task-specific knowledge. Another limitation is the lack of task-specific fine-tuning in the pre-trained BERT model before merging, which likely contributed to the poor performance. Future research could focus on fine-tuning both models on the target task, exploiting the strengths of each model more effectively. Furthermore, evaluating the merged model on a wider range of datasets could provide a more comprehensive understanding of its generalizability and performance across different NLP tasks. Finally, additional work could explore reducing the FLOPs and computational cost of these models without sacrificing performance, making them more efficient for real-world applications.

Most current studies in model merging primarily address scenarios where multiple models are fine-tuned on the same dataset or follow a similar training trajectory, often with different hyperparameter settings. However, there is a noticeable gap in the research when it comes to exploring the merging of models fine-tuned on distinct datasets, or models that have been trained from scratch on entirely different datasets (Yang et al. (2024a)). Future research could focus on developing a more rigorous theoretical framework that can guide the merging of models trained in diverse conditions (Yang et al. (2024a)). Such an analysis could improve the success rate of model merging and provide insights into the optimal conditions for achieving effective merges. A deeper understanding of these factors will not only enhance the reliability of model merging but also aid in identifying key prerequisites and configurations that promote better outcomes in this process.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, and OpenAI team. Gpt-4 technical report, 2023. URL https://arxiv.org/abs/2303.08774.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. Evolutionary optimization of model merging recipes, 2024. URL https://arxiv.org/abs/2403.13187.

Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, and Gemini Team. Gemini: A family of highly capable multimodal models, 2023. URL https://arxiv.org/abs/2312.11805.

Omri Avrahami, Dani Lischinski, and Ohad Fried. *GAN Cocktail: Mixing GANs Without Dataset Access*, page 205–221. Springer Nature Switzerland, 2022. ISBN 9783031200502. doi: 10.1007/978-3-031-20050-2_13. URL http://dx.doi.org/10.1007/978-3-031-20050-2_13.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

Stefan Horoi, Albert Manuel Orozco Camacho, Eugene Belilovsky, and Guy Wolf. Harmony in diversity: Merging neural networks with canonical correlation analysis, 2024. URL https://arxiv.org/abs/2407.05385.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding, 2020. URL https://arxiv.org/abs/1909.10351.

Chris McCormick. Glue benchmark (general language understanding evaluation) – explained, 2019. URL https://mccormickml.com/2019/11/05/GLUE/. Accessed on 24/09/2024.

Mohammed Muqeeth, Haokun Liu, and Colin Raffel. Soft merging of experts with adaptive routing. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=7I199lc54z. Featured Certification.

Dang Nguyen, Trang Nguyen, Khai Nguyen, Dinh Phung, Hung Bui, and Nhat Ho. On cross-layer alignment for model fusion of heterogeneous neural networks, 2023. URL https://arxiv.org/abs/2110.15538.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=0A9f2jZDGW.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020. URL https://arxiv.org/abs/1910.01108.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding, 2019. URL https://arxiv.org/abs/1804.07461.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models, 2023. URL https://arxiv.org/abs/2306.01708.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities, 2024a. URL https://arxiv.org/abs/2408.07666.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning, 2024b. URL https://arxiv.org/abs/2310.02575.