Generalizability of Mixture of Domain-Specific Adapters from the Lens of Signed Weight Directions and its Application to Effective Model Pruning

Tuc Nguyen
Department of Computer Science
Indiana University
nguyentuc1003@gmail.com

Thai Le
Department of Computer Science
Indiana University
tle@iu.edu

Abstract

Several parameter-efficient fine-tuning methods based on adapters have been proposed as a streamlined approach to incorporate not only a single specialized knowledge into existing Pre-Trained Language Models (PLMs) but also multiple of them at once. Recent works such as AdapterSoup propose to mix not all but only a selective sub-set of domain-specific adapters during inference via model weight averaging to optimize performance on novel, unseen domains with excellent computational efficiency. However, the essential generalizability of this emerging weight-space adapter mixing mechanism on *unseen, in-domain examples* remains unexplored. Thus, in this study, we conduct a comprehensive analysis to elucidate the generalizability of domain-specific adapter mixtures in in-domain evaluation. We also provide investigations into the inner workings of the mixture of domain-specific adapters by analyzing their weight signs, yielding critical analysis on the negative correlation between their fraction of weight sign difference and their mixtures' generalizability. The code is available at Github.

1 Introduction

Recently, several *parameter-efficient fine-tuning methods that are based on adapters* have been introduced as a streamlined approach for fine-tuning Pre-trained Language Models (PLMs) to equip them with new, specialized knowledge or domain. Several algorithms have been proposed to train a distinct adapter for each new domain (Houlsby et al., 2019; Pfeiffer et al., 2021; Hu et al., 2022). To further improve a model's generalizability, existing works (Pfeiffer et al., 2021; Wang et al., 2021a; Diao et al., 2023) mostly focus on training multiple adapters for multiple tasks and continuously adding more adapters for incoming new tasks. This can be inefficient for the new domain tasks that have only a few examples, making the learning among the tasks unequal. Thus, more recent works such as Matena and Raffel (2022); Wang et al. (2022, 2021b); Li et al. (2022); Chronopoulou et al. (2023) opt for weight-space averaging of model and/or adapters trained on different domains, resulting in so-called *Mixture of Expert Adapters*.

One recent notable work in this space is AdapterSoup (Chronopoulou et al., 2023), which proposes to merge the weights of a fixed-size, selective subset of different domain-specific adapters via an averaging function to accommodate unseen tasks or domains during inference. Such weight-space merging mechanism on adapters is efficient in practice as one can efficiently train a small, additional adapter and plug it into existing PLMs to incorporate new knowledge. Although the work reported favorable evaluation results on unseen, novel domains, it is unclear to what extent such weight-space merging mechanism on domain-specific adapters can generalize in an in-domain evaluation setting–i.e., how well it makes predictions on unseen examples of domains already seen during training. Moreover, to the best of our knowledge, no existing works comprehensively study the generalization of the mixture of adapters in the in-domain setting. This literature gap seems counter-intuitive because in-domain evaluation is fundamental and should precede out-of-domain evaluation. Moreover, in real-world applications, model owners have incentives to utilize as much as possible available information to improve their models over time. With the availability of parameter-efficient finetuning methods that are fairly easy to adopt with minimal space and runtime cost, the model owners are then incentivized to quickly fine-tune their models on a few examples collected from a new domain on an additional adapter to optimize the performance (rather than totally relying on out-of-domain prediction capability). As a result, although in-domain evaluation seems trivial, it is fundamental as one must ensure that the mixture of adapters works well on the tasks they have already

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 12956–12973

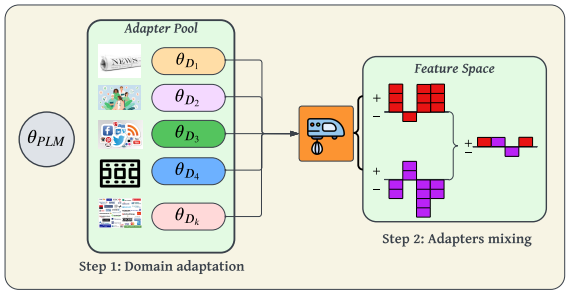

Figure 1: Mixing the adapter weights across various tasks may result in the importance weights of individual tasks nullifying each other, thereby yielding a merged mixture losing important information.

been trained on. Furthermore, several key questions regarding the resulting mixtures of domain-specific adapters remain unanswered, especially those regarding their generalizability and their adversarial robustness when mixing adapters trained from very different tasks.

Therefore, borrowing the pop-culture saying that *"mixed drinks and cocktails aren't actually the same thing"*, in contrast from existing works, we hypothesize that *not all* mixture of expert adapters are created equal and all have superior performance. Then, through an array of comprehensive experiments, we attempt to give answers to questions about *when and what to mix* when it comes to domain-specific adapters. We found that the weight-space merging mechanism suffers from performance degradation in terms of both generalizability and adversarial robustness even with inputs from domains it already trains on. Moreover, we also attempt to explain such performance degradation by revealing a critical negative correlation between *signed directions of adapter weights during mixing* and domain-specific predictive performance (Fig. 1). Although simple, this intuitive and novel observation also allows us to select *"when and what adapters to mix?"* and design a more effective model pruning as a by-product application.

Overall, our study does *not* focus on proposing a new mechanism, algorithm, or method. Instead, we focus on analyzing and bringing understanding of an existing and emerging paradigm of mixing multiple domain-specific adapters that was previously introduced (Chronopoulou et al., 2023; Wang et al., 2022). Specifically, we focus on in-domain prediction when mixing adapters from different domains as an emerging and potential paradigm for the deployment of PLMs in practice.

Our contributions are summarized as follows.

1. This is the first and most comprehensive analysis of in-domain generalizability of a mixture of domain-specific adapters with 3 different adapter methods on 13 diverse classification datasets,
2. We provide insights and analysis on when and what adapters to mix to minimize performance degradation via the lens of signed directions of adapters' parameter weights,
3. We demonstrate the utility of such insights to train mixtures of adapters with 90% sparsity that improve both generalizability and efficiency.

2 Related works

Adapter Fine-tuning. The primary method for adapting general-purpose PLMs to downstream tasks is via *full fine-tuning*, which requires adjusting all models' parameters (Peters et al., 2018; Devlin et al., 2019a). However, this results in redundant copies of fine-tuned models for each task, posing a significant memory challenge. Thus, various *parameter-efficient fine-tuning methods* have been proposed, including prompt-based (Li and Liang, 2021) and adapter-based fine-tuning (Houlsby et al., 2019; Pfeiffer et al., 2021; Hu et al., 2022). Among adapter-based fine-tuning methods, Houlsby (Houlsby et al., 2019) introduces two adapter blocks with bottleneck networks in each Transformer block of a PLM. Similarly, the Pfeiffer (Pfeiffer et al., 2021) adapter differs in architecture, incorporating only one adapter layer in each Transformer block, in contrast to the two layers introduced by Houlsby (Houlsby et al., 2019). LoRA (Hu et al., 2022) takes a distinctive approach by freezing the MLP modules of transformers and representing updates to attention weights with two low-rank matrices to optimize space while effectively retaining model performance. In this work, we focus on analyzing adapter-based fine-tuning methods as they are more popular and effective.

Mixture of Expert Adapters. Additionally, several approaches (Wang et al., 2021a; Pfeiffer et al., 2021, 2020; Wang et al., 2022) have been proposed to further optimize their adapters for various downstream tasks by maintaining a set of adapters and combine them during inference. Particularly, AdaMix Wang et al. (2022) fine-tunes so-called Mixture of Experts (MoEs) with adapters on a downstream task and averaging their weights during inference. In addition, Li et al. (2022) explores performance in novel domains through weight averaging on entire language models. Similarly,

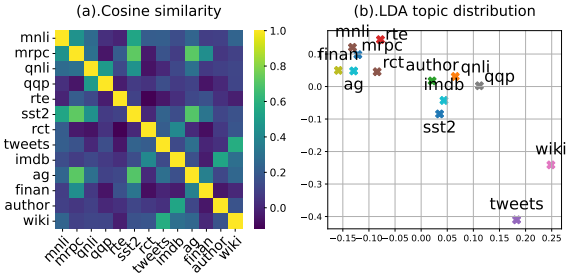

Figure 2: (a) Datasets' semantic similarity via cosine-similarity among centroids of Universal Sentence Encoder (USE) (Cer et al., 2018) embeddings of 1K randomly sampled documents from each dataset. (b) Topic distributions via Latent Dirichlet Allocation (LDA) (Blei et al., 2003).

AdapterSoup (Chronopoulou et al., 2023) opts for weight-space averaging of adapters trained on different domains. Among these methods, weight-space averaging is identified as the most intuitive method for mixing different adapters (Jin et al., 2023a) and (Chronopoulou et al., 2023).

Nevertheless, none of these works comprehensively evaluates and analyzes the generalizability and adversarial robustness of the resulting mixture of adapters under different mixtures of domain-specific knowledge, which is necessary to answer the question *"when and what to mix?"*.

3 Comprehensive In-Domain Evaluation

To evaluate the in-domain performance of adapter mixtures, we train several adapters with domain-specific knowledge and mix them in different combinatorial combinations. Then, we evaluate each combination on different downstream tasks on two aspects: (1) *generalizability* on unseen in-domain examples and (2) *adversarial robustness* under adversarial text attacks.

3.1 Evaluation Datasets

Diverse Domain Knowledge. To simulate knowledge diversity, we gather a total of 13 distinct and diverse *domain-specific* datasets of classification tasks for evaluation. We refer the readers to Appendix A.1 for detailed information and their linguistic statistics. Fig. 2 reveals the intricate diversity within our selected datasets, both semantic and topic-wise. Notably, SST2 and IMDB, both originating from the same movie corpus, exhibit proximity in topic embedding spaces. On the contrary, non-formal datasets such as Wiki and Tweets are distinctly distant from other datasets in this regard. We refer the readers to Appendix. A.2 for a detailed exploration and analysis of the topic distributions among the datasets.

3.2 Mixing Fine-Tuned Adapters

Base Models and Individual Adapters. We design our evaluation using two transformer-based models, namely BERT (Devlin et al., 2019b) and RoBERTa (Liu et al., 2019), with a 3 diverse and well-known adapter methods. They are Houlsby (Houlsby et al., 2019), Pfeifer (Pfeiffer et al., 2021) and LoRA (Hu et al., 2022). These adapter-based methods introduce variations in the adapter architecture and parameterization (Sec. 2), contributing to the comprehensiveness of our analysis.

Mixing Adapters. From the pre-trained weights θ_{PLM} of either BERT and RoBERTa, we train a suite of 13 domain-specific adapters tailored for diverse domains, denoted as $\theta_{D_1}, \theta_{D_2}, \ldots, \theta_{D_k}$. Following (Chronopoulou et al., 2023), the final inference of the target mixture of domain-specific adapters becomes:

$$f(x, \theta_{PLM} + \frac{1}{k}\sum_{i=1}^{i=k} \theta_{D_i}) \quad (1)$$

3.3 Adversarial Text Generation

Textual adversarial attacks are popular in AI robustness research. Given a dataset $D=\{(x_i, y_i)\}_i^N$, where x represents the sample and y denotes the ground truth label, a textual adversarial attack aims to attack a PLMs f_θ with a classification loss function $\mathcal{L}$ by perturbing each sample x with perturbation noise δ given a certain budget C:

$$\arg\max_{\delta \in C} \mathcal{L}[f_\theta(x+\delta), y], \quad (2)$$

Toward evaluating the robustness of a mixture of adapters, we employ *both black-box and white-box* textual attacks to exercise Eq. 2. We utilize the popular *TextFooler* (Jin et al., 2020) as the *black-box attack*, which aims to replace words with synonyms or contextually similar words to deceive PLMs. We utilize the well-known *FGSM* (Goodfellow et al., 2015) as the *white-box attack*, which can efficiently craft adversarial examples by perturbing embedding of text data in the direction of the sign of the gradient of the loss function to the input, thereby exposing vulnerabilities in model robustness.

3.4 Combinatory Evaluation

To evaluate in-domain performance for each target domain, we generate all possible combinations of adapters. To illustrate, for the target domain MNLI, we can first evaluate with a mixture of only

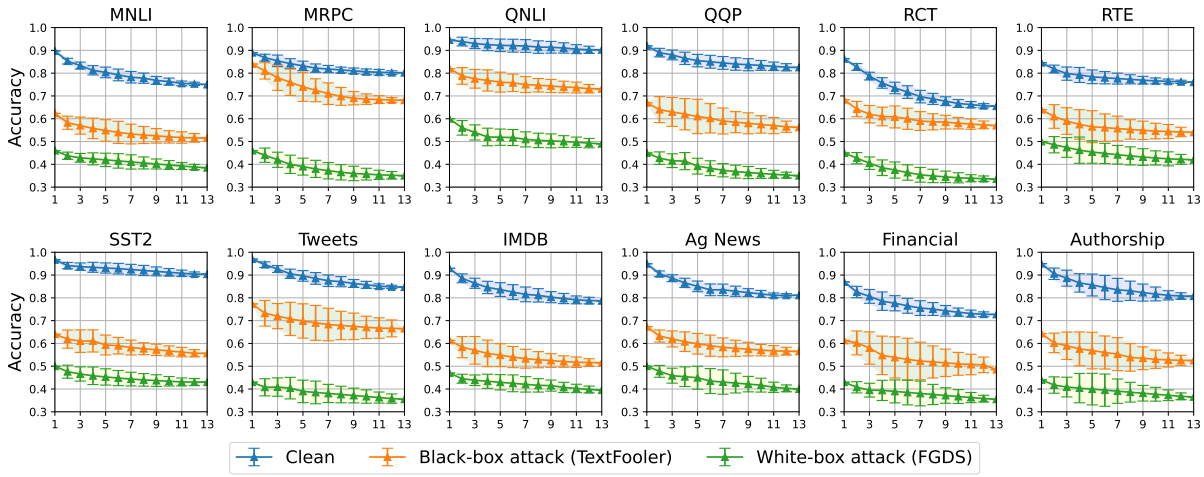

Figure 3: Accuracy of RoBERTa with Pfeiffer (Pfeiffer et al., 2021) in each target domain. *X-axis denotes the number of mixed adapters*.

Dataset	mnli	mrpc	sst2	rte	qnli	qqp	rct	ag	authorship	financial	imdb	tweets	wiki	Average
∇_{clean}	**15.04**	8.31	4.90	8.50	4.19	9.92	**21.72**	**13.69**	12.60	**15.33**	**14.26**	**13.17**	**13.16**	11.91
$\nabla_{blackbox}$	12.10	**16.07**	**10.43**	**10.21**	10.13	9.82	12.32	12.62	**13.42**	13.82	13.04	11.83	10.52	**12.03**
$\nabla_{whitebox}$	10.14	11.21	8.31	9.42	**12.14**	**10.24**	12.53	12.06	10.45	9.53	10.04	10.24	7.53	10.30

Table 1: Average *absolute* performance drop (*in percentage %*) of RoBERTa with Pfeiffer (Pfeiffer et al., 2021) when mixed from all domain adapters on clean, black-box, and white-box attacks.

itself. When combining two adapters, we have the flexibility to choose 1 additional adapter out of the remaining 12, resulting in 12 possible combinations. For a set of 3 adapters, including MNLI, we select 2 adapters out of the 12 to generate C_{12}^2 combinations. This process continues for sets ranging from 4 to 13 adapters, where, in the case of 13 adapters, all adapters are combined. Thus, we have $13 * (1+\sum_{i=1}^{12} C_{12}^i)=53,248$ combinations for all domains. We report *mean and variance* of in-domain performance for each set of mixtures of k adapters.

Notably, this setup already *includes all the possible mixtures* potentially selected by AdapterSoup (Chronopoulou et al., 2023), which proposes an additional mechanism to select a subset of a fixed number of domain-specific adapters to mix.

4 Experiments

4.1 Implementation Details

For the Ag News, Authorship, Financial, IMDB, Tweets, and Wiki-Toxic, we partition the dataset into three segments with an 8:1:1 for train:val:test splits. For datasets belonging to the GLUE corpus, we employ their public training and evaluation splits. For the black-box *TextFooler* attack, we set the minimum embedding cosine similarity between a word and its synonyms as 0.8, and the minimum USE similarity is 0.84. For white-box FGSM (Goodfellow et al., 2015) attack, we set the perturbation magnitude to 0.01. Following the setup of Houlsby and Pfeiffer (Houlsby et al., 2019; Pfeiffer et al., 2021), we use all adapters with a dimension of 64 and 256 for RoBERTa-large and BERT-base models. With LoRA, we use rank r=4 following (Hu et al., 2022). Detailed training, evaluation dataset, and hyper-parameter configuration for different tasks are presented in Appendix A.3.

4.2 In-Domain Performance Results

Overview. We present the performance of a candidate setting of RoBERTa with Pfeiffer adapter (Pfeiffer et al., 2021) with different numbers of additional mixing domains in Fig. 3. Table 1 shows how much the predictive performance drops without and with adversarial black-box and white-box attacks when mixing all adapters. Overall, the average performance drops over all tasks on the clean test set from the original performance to a mix of all adapters is 11.91%, and that for black-box and white-box attacks are 12.03% and 10.30%, respectively. Further results of *generalization and adversarial robustness performance across other models and adapter methods* are documented in Appendix A.4.

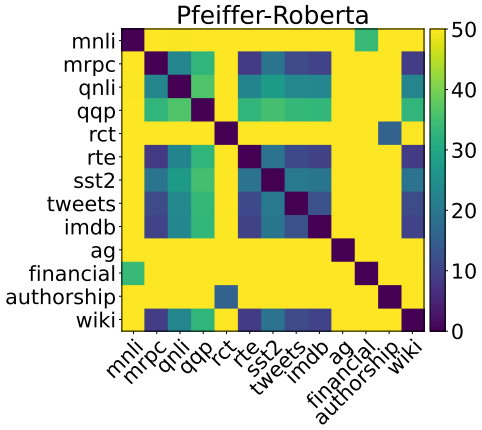

Figure 4: Heatmap visualization of the Fraction of Sign Difference (in %) of Pfeiffer Adapters (Pfeiffer et al., 2021) trained on 13 domain-specific tasks with RoBERTa..

Finding #1: As we add more tasks or domains, the predictive performance of every single task decreases, reaching its lowest point when we incorporate the maximum of 13 adapters training on various topic datasets. Fig. 3 shows that mixing domain-specific adapters indeed decreases in-domain performance (reduction of around 10% in MRPC, QQP, RTE, etc., and nearly 17% in the Financial domain when mixing 13 adapters). The same behaviors were also observed in (Jin et al., 2023b) where they merge the weight of PLMs. Notably, task accuracies decreased at a slower pace for QNLI and SST2 when evaluating with increasing size of mixtures (Fig. 3). In contrast, a substantial decrease in accuracy is observed for domains such as RCT, IMDB, Ag News, and Authorship (Fig. 3). This shows that mixing domain-specific adapters impair model performance differently depending on the target domain, or *"what to mix" in an adapter mixture has a crucial effect on the mixture's performance.* To attempt to explain this behavior, we later present and verify a hypothesis that such mixing domain-specific adapters via weight averaging can result in "forgotten knowledge" that can happen due to the differences in signs when mixing these adapters (Sec. 5).

Finding #2: On average, there are no notable differences of the magnitude in accuracy degradation with and without adversarial attacks (12.03% dropped in black-box attack versus 11.91% dropped without attack) (Fig. 3). The overall predictive performance was significantly lower under the white-box compared to the black-box attack as expected because the white-box FGSM attack has additional access to the models' parameters. Interestingly, on the RCT dataset, the accuracy drop on clean (21.72%) is much larger than on black-box and white-box attacks (12.32% and 12.53%, respectively).

Finding #3: Variance in task accuracy on adversarial attacks, when combining different domain adapters, is observed to be larger than the variance in clean accuracy (Fig. 3). Although the magnitude decrease appears similar in Fig.3, differences in variance (max, min) are discernible among each combination. Specifically, in MRPC, QNLI, QQP, RTE, SST2, Tweets, IMDB, Financial, and Authorship, we can observe that certain mixed models observed slightly better adversarial robustness compared to single adapters (when the number of mixing adapters k is 2 or 3). Moreover, the variance of robustness in adversarial scenarios tends to be higher than in clean scenarios because all of the adapters are trained on different tasks and they may exhibit different vulnerabilities to the same attack method.

Finding #4: Mix up to three adapters to maintain competitive performance. Based on the results from Fig.3 and all findings above, *it is advisable to mix only up to three tasks* to maintain competitive performance (as observed in MRPC, QQP, RTE, Tweet, Financial, and Authorship domains with less than a 3% accuracy drop in accuracy). Especially, in some cases when evaluating QNLI, SST2, mixtures of less than three adapters even achieved better performance in terms of generalizability (up to 1%) and also in terms of adversarial robustness of up to 3% compared to the original performance.

5 Effects of Sign Differences of Adapter Weights during Mixing: A Hypothesis

5.1 An Explanation Hypothesis

The ideal scenario when averaging adapter weights during mixing is to make minimal adjustments to their weights, both in terms of values–i.e., magnitudes, and directions, to sustain as much as possible the knowledge learned. Investigating how adapter weights mix regarding both their magnitudes and directions, can be overly complex. Thus, we simplify this assessment by focusing on the *sign directions* of the adapter weights in our analysis. Following the mixing process of k individual adapter weight in Eq. 1 (Sec. 4), we then hypothesize that mixing adapter weights of conflicting signs can result in "forgotten knowledge", and lead to performance degradation. As illustrated in Fig. 1, averaging adapter weights across various tasks may

lead to nullifying importance weights for individual tasks if their signs are opposite–i.e., positive v.s. negatives. In other words, the **fraction of sign difference or FSD (%)** or proportion of weight sign difference in adapter weights during mixing correlate with their mixtures' generalizability. Alg. 3 (Appendix A.5) shows the calculation of FSD.

We evaluate our hypothesis with different cases: (i) individual adapters (mixture with k=1), (ii) dual adapters (k=2), and generalize to (iii) multiple adapters (k>2). To demonstrate the utility of our hypothesis, we apply it to improve the generalizability of adapter mixtures and also to derive a more effective model pruning in Sec. 6, 7.

5.2 Individual Adapters (k=1)

We calculate the FSD of adapter weights on RoBERTa and normalize it by the total number of weights, denoted as a matrix $\mathbf{S}_{k\times k}$ where each row is the FSD of a single adapter train on task k to the remaining adapters (Fig. 4). We refer the readers to Sec. A.6 in the Appendix for results on BERT. Interestingly, a consistent trend in the FSD is observed across various model architectures (BERT, RoBERTa) and adapter methods (e.g., Fig. 4 and Fig. 9 of Appendix A.6). The reason is that adapters act as small MLP layers that integrate task-specific knowledge into pre-trained models (Meng et al., 2022) in different adapter methods. This shared functionality contributes to a similar trend in FSD, highlighting the robustness and generalizability of the observed behavior across different adapters' architectural variations. In addition, Adapters trained on datasets with distinct topic distributions and cosine similarities (Fig. 2) exhibit varying weight directions (Fig. 4). Especially, MNLI has a similar linguistic distribution with other datasets (i.e. MRPC, RTE, Ag News, etc) (Fig. 2), but the adapter trained on MNLI has a significantly larger FSD compared to the remaining domain-specific adapters (Fig. 4). Thus, *datasets that are similar in linguistic statistics may not necessarily share the same optimization trajectory*. As a result, methods that are based on the linguistic distribution to choose the closest set of adapters to mix like AdapterSoup (Chronopoulou et al., 2023) may lead to sub-optimal performance.

5.3 Dual Adapters (k=2)

Fig. 5 shows the FSD for each adapter when mixing two domain-specific adapters. This investigation is conducted within the context of the

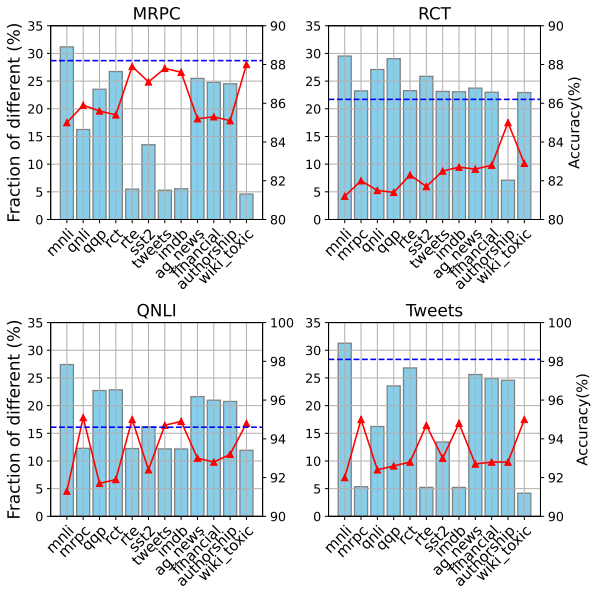

Figure 5: FSD when mixing two (k=2) adapters. Sky-blue bars show the FSD (left y-axis). *Dashed blue lines* denote the accuracy achieved by a standalone adapter. *Solid red lines* illustrate the variations in accuracy after mixing. Please refer to Fig. 14 in Appendix A.7 for results in other tasks.

Pfeiffer Adapter (Pfeiffer et al., 2021) using a pre-trained RoBERTa model. Overall, there is a strong negative correlation between the FSD and the generalizability–i.e., the lower the sky-blue bar (or the smaller fraction of weight sign conflicts), the higher the performance of the mixture (Fig. 5). Notably, *tasks with substantial difference in the weight signs witness a pronounced performance decrease*. Specifically, RCT exhibits significant performance drops due to substantial differences in adapter weight direction. Conversely, tasks such as MRPC, and QNLI demonstrate either marginal improvement or no change in performance when mixed with other adapters. This is well correlated to the marginal FSD of the mixed adapters, ranging from only 5% to 10% compared to the original weight.

5.4 Multiple Adapters (k>2)

Similar to the dual-adapter setting, there is *still a strong negative correlation between FSD and the generalizability*, and increasing the number of mixed adapters amplified the sign disparity (Fig. 6). For example, adapters trained on RCT and Tweet exhibit large FSD compared to other adapters and hence observed a significant decrease in generalizability (Fig. 6). In some cases, mixing only a few adapters (e.g., 2–4) could still maintain competitive performance as in QNLI, SST2 domains (Fig. 6, 15). This correlates with the relatively

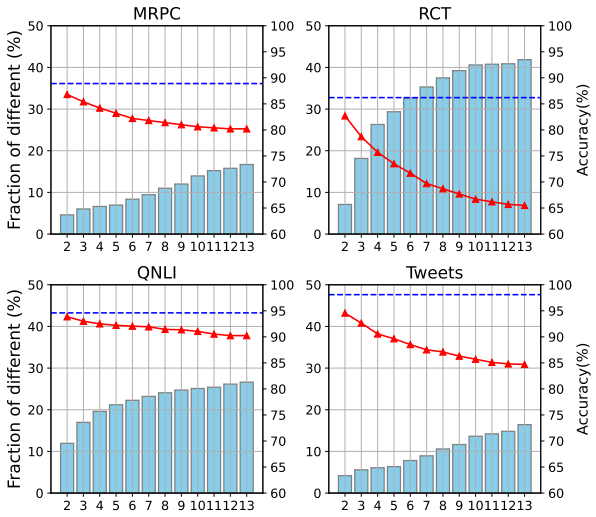

Figure 6: Fraction of weights changing direction during the mixing of multiple adapters, ranging from 2 to 13. The chart description is similar to Fig. 5. We refer to Fig. 15 in Appendix A.7 for detailed results in other tasks.

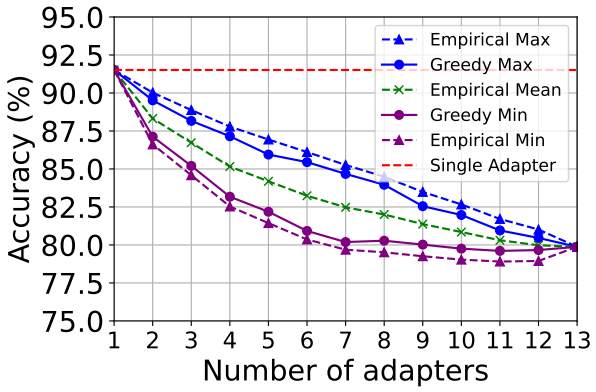

Figure 7: Average model accuracy of 13 domains under different numbers of mixed adapters (l) using the guidance of FSD.

marginal differences in adapter weights of QNLI and SST compared to adapters trained on other domains (Fig. 4), as their mixtures do not lead to significant cancellations of existing parameters and hence preserve learned knowledge.

5.5 Discussion

Which adapters we should mix? From Sec. 5, we find that it is not advisable to merge adapters that are significantly different from each other in weight signs. On the other hand, mixing the weight of these adapters with other adapters which have a small FSD achieves competitive performance (within 3% drop in accuracy) only when the number of mixing adapters is small (QNLI, MRPC in Fig. 6) or can achieve better performance, although rarely, compared to training the original adapter as seen in QNLI domain (Fig. 5). Therefore, when deploying PLMs, it is prudent to only select a group of tasks with a small FSD to minimize the performance drop in the final mixed model. Our observation is crucial in deploying these models in edge devices where only the adapters are saved on edge, which often has a specific memory capacity limit.

Algorithm 1: Greedy Adapter Mixing

Input: k domain-specific adapters, a matrix $\mathbf{S}_{k\times k}$ of FSD, l number of adapters to fusion.
Output: Average(candidates)

```
candidates ← {}
Compute the average of FSD
avg_S = mean(S, axis = 1)
Select the top l from set of k adapters according to
smallest average FSD
candidates ← top_l
return average(candidates)
```

Experiment comparison with AdapterSoup. AdapterSoup (Chronopoulou et al., 2023) dynamically selects a set of l adapters during inference. When l=k, then our experiment setting is the same as the AdapterSoup setting when we use all adapters at once. When l=1, it is the original performance of a single adapter, assuming that AdapterSoup is perfect at picking the same domain that is already trained on, and this result is already included in our paper (mixture of only 1 adapter, Fig. 3). When 1<l<k, we do not have the results of the specific combination that AdapterSoup would select. However, we reported the maximum performance across all combinatorial combinations among 13 domain-specific adapters or each value in Fig. 3. For example, when l=3, we only observe a possible comparative performance (within less than 3% drop) in QNLI, MRPC, and SST2 domains (Fig. 15). We also emphasize that it is one thing to select the best adapters to mix during inference, it is much harder to choose l or how many of them.

6 Greedy Adapter Mixing with FSD

Our observations from Sec. 5 reveal that two adapters with minimal disparity in FSD can yield competitive performance when combined. Therefore, to demonstrate the utility of FSD, in this section, we design a mixing strategy, so-called *Greedy Adapter Mixing* (Alg. 1), that utilizes FSD to decide which domain-specific adapters to mix by minimizing the overall FSD in a greedy manner to get a final mixed model with competitive performance. This algorithm is also based on the hypothesis that mixing adapters with minimal FSD can yield mixture models of better generalizability. We proceed

by evaluating model performance across various adapter combinations.

Fig. 7 shows the generalizability performance of a mixture of $l \in [2, 13]$ domain-specific adapters. Overall, Greedy Adapter Mixing resulted in very competitive performance compared with empirical upper-bound accuracy. In contrast, using the same algorithm but maximizing FSD resulted in performance close to the empirical lower-bound accuracy (Fig. 7). However, in both two cases, mixing adapters with FSD cannot achieve the empirical upper-bound and lower-bound performance. Thus, *greedily mixing adapters to minimize FSD does not totally prevent knowledge loss in the adapter mixtures*. Nevertheless, FSD is still useful as a guidance measure to effectively mix domain-specific adapters.

7 Towards Effective Model Pruning

To further demonstrate the utility of our FSD analysis in Sec. 5 and Sec. 6, in this section, we leverage FSD information to reduce knowledge loss through the development of a pruning algorithm guided by FSD insights. Specifically, Fig. 5 shows that predictive performance experiences a significant drop when integrating adapters with pronounced disparities in weight signs–i.e., positive v.s. negative signs. Moreover, neural network pruning indicates that only a limited number of weights significantly contribute to task performance, suggesting redundancy within the weights that can be pruned without compromising the original task performance (Han et al., 2016; Frankle et al., 2021; Lazarevich et al., 2021). Thus, to mitigate the impact of weight sign differences in adapter mixtures, we propose mixing only the *sparse versions* of the adapters' weights.

Different from Alg. 1, this strategy *indirectly reduces the fraction of weight sign conflicts*. Intuitively, by minimizing the FSD, the mixing process becomes more resilient to the inadvertent elimination of important weights by less significant or redundant weights. This phenomenon is visually depicted in Step 2 of Fig. 1, where only significant weights in the two adapters need to be preserved, and small or unimportant weights of opposing signs can be eliminated.

7.1 FSD-based Magnitude Pruning

Post-training Pruning. Sparse Adapter (He et al., 2022) employs pruning across every layer of adapters, being able to achieve comparable or even superior predictive performance than standard adapters, even when the sparse ratio reaches up to 80%. By adopting a similar process of adapters pruning after training, *with the guidance of our FSD analysis in Sec. 5*, we can *eliminate redundant parameters at an early stage, circumventing the need for a time-consuming iterative pruning process*, as discussed in prior works such as Han et al. (2016); Lazarevich et al. (2021). *The detailed pruning algorithm is presented in Alg. 2.*

Algorithm 2: FSD-based Pruning

Input: adapter parameters w, sparse ratio s.
Output: pruned adapter $\tilde{w}$.

$w \leftarrow \text{Trained}(w)$
Compute important score $z = |w|$
Compute the s-th percentile of z as z_s
$m \leftarrow \mathbb{1}\left[z - z_s \geq 0\right]$
$\tilde{w} \leftarrow m \odot w$

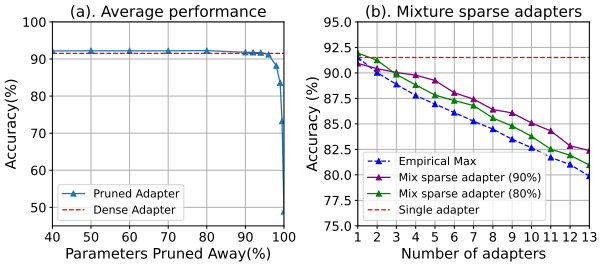

Figure 8: (a) Average RoBERTa performance with a single sparse adapter across 13 domains with increasing sparsity. (b) Model accuracy when increasing the # of sparse adapters being mixed. *Dashed red* lines represent model generalization when mixing domain-specific adapters. *Dashed blue* lines depics the maximum performance when mixing k domain-specific dense adapters. *Solid green* and *solid purple* lines represent model performance when mixing k domain-specific adapters with 80% and 90% sparsity, respectively.

Generalizability of Pruned Adapters. Fig. 8a shows RoBERTa's performance, where we systematically prune the weight of the Pfeiffer adapter 40%–100% of sparsity. For a single task at the sparsity level $d\%$, we retain only the largest–i.e., the top-$d\%$, influential parameters of the corresponding adapter, and report the average in-domain performance across 13 domains. Remarkably, pruning up to 90% parameters of adapter weight does not lead to performance degradation. This observation suggests redundancy in adapters' parameters may *contribute to the increase in the fraction of weight direction conflicts*–i.e., high FSD when merging them (Fig. 6).

Generalizability of Mixed Pruned Adapters. Motivated by pruned adapter can still maintain original performance up to 90% of sparsity (Fig.

8a), we hypothesize that mixing sparse adapters may indirectly reduce the fraction of weight sign conflict, therefore, leading to competitive performance with the original adapters. Given a set of domain-specific adapters, based on the FSD, we choose the top k layers with the highest fraction of weight sign difference and prune each layer up to 90% of sparsity. Then we mix these adapters by weight averaging. Details of mixing domain-specific adapters with weight sign conflict information is shown in Alg. 4 (Appendix A.8). Fig. 8b shows that mixing adapters with 80% or 90% sparsity consistently achieved better performance than the upper-bound empirical accuracy achieved when mixing their dense versions.

8 Conclusion

This work provides a comprehensive empirical in-domain evaluation of the emerging mechanism of mixing domain-specific adapters. We also provide insights into the inner workings of the mixture of domain-specific adapters by analyzing their weight signs, yielding critical observations on the negative correlation between the fraction of sign difference among adapters and their mixtures' generalizability. By examining the signed directions of adapter weights, we also offer the readers valuable advice on the optimal selection of adapters to mix to achieve competitive performance. Such examination also helps enhance our understanding of the interconnected role of weight sign difference in the context of sparse neural networks.

Limitation

Primarily, our exploration focused solely on one classic pruning method, namely Magnitude Pruning (Sanh et al., 2020) while there are existing more advanced pruning techniques such as SynFlow (Tanaka et al., 2020), GraSP (Wang et al., 2020) that are also applicable for condensing neural network architectures. Consequently, future works include investigating the applicability of our findings to these alternative pruning approaches. Furthermore, our examination was confined to the natural language understanding tasks. A valuable avenue for future research would involve extending our analysis to encompass the emerging text generation tasks, particularly within the context of the current transformer-based language model, including but not limited to the machine translation tasks utilizing complex GPT-family models.

References

Malik Altakrori, Jackie Chi Kit Cheung, and Benjamin CM Fung. 2021. The topic confusion task: A novel evaluation scenario for authorship attribution. In *EMNLP*.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. In *TAC*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. In *Journal of machine Learning research*.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder for english. In *EMNLP*.

Alexandra Chronopoulou, Matthew E Peters, Alexander Fraser, and Jesse Dodge. 2023. Adaptersoup: Weight averaging to improve generalization of pre-trained language models. In *EACL*.

Franck Dernoncourt and Ji Young Lee. 2017. PubMed 200k RCT: a dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019b. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Shizhe Diao, Tianyang Xu, Ruijia Xu, Jiawei Wang, and Tong Zhang. 2023. Mixture-of-domain-adapters: Decoupling and injecting domain knowledge to pre-trained language models memories. In *ACL*.

Bill Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing*.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. 2021. Pruning neural networks at initialization: Why are we missing the mark? In *ICLR*.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

Song Han, Huizi Mao, and William J Dally. 2016. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*.

Shwai He, Liang Ding, Daize Dong, Jeremy Zhang, and Dacheng Tao. 2022. SparseAdapter: An easy approach for improving the parameter-efficiency of adapters. In *EMNLP*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *ICLR*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *ICLR*.

Shankar Iyer, Nikhil Dandekar, Kornél Csernai, et al. 2017. First quora dataset release: Question pairs. In *data. quora. com*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *AAAI*.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023a. Dataless knowledge fusion by merging weights of language models. In *ICLR*.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. 2023b. Dataless knowledge fusion by merging weights of language models. In *ICLR*.

Ivan Lazarevich, Alexander Kozlov, and Nikita Malinin. 2021. Post-training deep neural network pruning via layer-wise calibration. In *ICCV*.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A Smith, and Luke Zettlemoyer. 2022. Branch-train-merge: Embarrassingly parallel training of expert language models. In *arXiv*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. In *arXiv*.

Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. In *Journal of the Association for Information Science and Technology*.

Michael Matena and Colin Raffel. 2022. Merging models with fisher-weighted averaging. In *NeurIPS*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. In *NeurIPS*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *EACL*.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *EMNLP*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *EMNLP*.

Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. In *NeurIPS*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. 2020. Pruning neural networks without any data by iteratively conserving synaptic flow. In *NeurIPS*.

Chaoqi Wang, Guodong Zhang, and Roger Grosse. 2020. Picking winning tickets before training by preserving gradient flow. In *ICLR*.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2021a. K-adapter: Infusing knowledge into pre-trained models with adapters. In *ACL*.

Xinyi Wang, Yulia Tsvetkov, Sebastian Ruder, and Graham Neubig. 2021b. Efficient test time adapter ensembling for low-resource language varieties. In *EMNLP*.

Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. 2022. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In *EMNLP*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL*.

A Appendix

Dataset	mnli	mrpc	sst2	rte
Train	392,702	3,668	67,349	2,490
Test	9,815	408	872	277
Dataset	**qnli**	**qqp**	**rct**	**ag**
Train	104,743	363,846	178,882	120,000
Test	5,463	40,430	30,135	7,600
authorship	**financial**	**imdb**	**tweets**	**wiki**
2,743	4,846	22,500	31,962	127,656
686	484	2,500	3,196	63,978

Table 2: Number of instances for each dataset divided by training and test set.

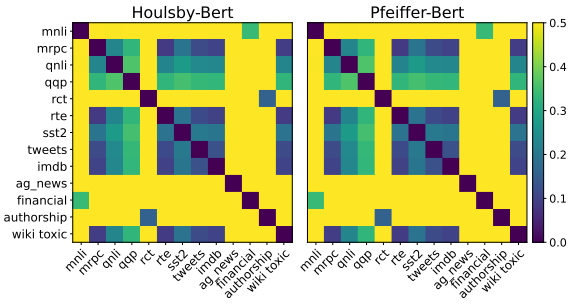

Figure 9: Fraction on differences of adapter weight direction.

Algorithm 3: Fraction of weight sign difference (FSD).

Input: two adapters with similar architecture learned from two domain θ_A, θ_B
Output: Fraction f of weight sign difference

Compute total parameters s in each adapter ($s = s_A = s_B$)
$C \leftarrow \{\}$
For every layer in adapter θ_A, θ_B, compute
element-wise product for each layer.
for k, v **in** $\theta_A.items()$ **do:**
  $C[k] = \text{mul}(\theta_A[k], \theta_B[k])$
Count total of numbers which value is
smaller than 0.
for k, v **in** $C.items()$ **do:**
  $counter += \text{sum}(value < 0)$
$f = counter/s$
return f

A.1 Datasets.

Diverse Knowledge Datasets. To simulate knowledge diversity, we gather a total of 13 distinct and diverse *domain-specific* datasets or classification tasks for evaluation. They are MNLI (Williams et al., 2018), QNLI (Rajpurkar et al., 2016), RTE (Bentivogli et al., 2009), MRPC (Dolan and Brockett, 2005), QQP (Iyer et al., 2017) and SST2 (Socher et al., 2013) from the *GLUE domain* corpus. PubMed-20K RCT dataset (Dernoncourt and Lee, 2017) from *Biology domain* for sentence classification. IMDB dataset from a *Movie Review domain*. Ag News, Financial (Malo et al., 2014) and Guardian Authorship (Altakrori et al., 2021) are *News domain* datasets across World, Sports, Business, Science/Technology, and Financial topics. Wiki Toxic [1] and Tweets Hate Speech are two *Informal text domain* for toxicity detection.

Linguistic Statistic. Table 3 shows detailed statistics such as number of documents, average document length, and sentence lengths.

A.2 Topic distribution of training datasets

Tables from 4 to 16 show 10 topics and corresponding important words which are exacted from LDA for each training dataset. Notably, Ag News and SST2 have high cosine similarity but observe a large difference in terms of topic distribution compared to other domains (Fig. 2). Therefore, each statistical mechanism like cosine similarity or topic distribution only reflects one aspect of data distribution and may show inconsistencies with each other.

A.3 Hyper-parameter

Training and evaluation datasets. To assess performance in out-of-distribution scenarios, we conduct evaluations on a diverse set of 13 datasets covering various topics, ranging from movie reviews, news, authorship, and healthcare, to non-formal language text such as Wiki Toxic and Tweets. For datasets within the GLUE corpus, we employ training and evaluation datasets to gauge accuracy across different settings. In the case of Ag News, Authorship, Financial, IMDB, Tweets, and Wiki-Toxic, we partition the training set into three segments with an 8:1:1 ratio, utilizing them for training, evaluation, and test datasets, respectively. This approach ensures a comprehensive evaluation of model performance across a wide spectrum of domains and linguistic styles. Table 2 shows data statistics on train/test datasets.

Setting on text adversarial attack. In this study, we employ two types of attacker methods: *TextFooler* (Jin et al., 2020) and FGSM (Goodfellow et al., 2015).

TextFooler word-level attacks focus on replacing words within the text with synonyms or contextu-

[1] https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/

Data Source	Average Document Length	Average Sentence Length	Average # Sentences per Document
MNLI	15.1	14.7	1.0
MRPC	21.9	21.1	1.0
QNLI	18.2	18.0	1.0
QQP	11.1	9.9	1.2
RTE	26.2	18.1	1.4
SST	10.4	10.4	1.0
RCT	26.5	26.3	1.0
Ag-news	38.4	29.1	1.3
Authorship	1038.6	20.2	51.3
Financial	23.1	22.8	1.0
IMDB	233.8	21.6	10.8
Tweets	15.9	9.6	1.6
Wiki-toxic	67.8	15.4	4.4

Table 3: Length statistics.

Table 4: Topic distribution on MNLI dataset

#Topic	MNLI
1	well, time, got, take, one, much, day, something, ive, even, way, long, little, make, back
2	kind, system, though, come, went, well, today, view, church, including, president, seems, across, run, policy
3	say, get, cost, guess, were, business, car, local, whole, north, rather, getting, question, technology, capital
4	service, state, world, get, big, pretty, give, war, yes, standard, real, here, came, call
5	probably, high, thought, however, set, hand, enough, said, since, type, jon, yet, and, service
6	could, mean, around, part, another, change, percent, made, course, life, book, fact, name, room
7	government, program, federal, information, country, problem, le, new, national, may, number, agency, report, organization
8	year, two, house, case, old, three, town, street, century, one, city, study, man, four, different
9	know, like, think, thats, right, really, people, thing, good, go, one, lot, going
10	yeah, work, legal, rule, last, year, he, american, small, home, company, act, group, analysis, public

ally similar words. By making ostensibly minor alterations to the input text, these attacks can deceive LLMs into producing incorrect outputs or substantially modifying their predictions. We meticulously fine-tune the hyperparameters of TextFooler to obtain more appropriate synonyms. We set the minimum embedding cosine similarity between a word and its synonyms as 0.8, and the minimum Universal Sentence Encoder similarity is 0.84.

FGSM (Goodfellow et al., 2015) is a white-box embedding-level attack. FGSM uses the Fast Gradient Sign Method to calculate gradients of the model's loss to the input text and generates an adversarial example by perturbing the embedding of input text in the direction that maximizes the loss. We choose the magnitude of the perturbation in embedding space as 0.01 on BERT and RoBERTa models.

Adapter Configuration. We use adapters with a dimension of 64 and 256 using RoBERTa-large and BERT-base encoders following the setup of (Houlsby et al., 2019), (Pfeiffer et al., 2021). With LoRA, we use rank $r = 4$ following the setup of (Hu et al., 2022).

Hardware Information. We evaluate model performance on AMD Ubuntu 22.04.2 with Ryzen Threadripper PRO 5975WX, 1800MHz, Cached 512 KB and 4 $\times$ GPU Nvidia A6000. **Hyper-Parameters.** Detailed hyper-parameter configuration for different tasks is presented in Table 17.

A.4 Model performance when mixing adapters across tasks

Tables 10, 11, 12 and 13 show task accuracy when mixing multiple adapters.

A.5 Fraction of weight sign difference

Alg. 3 presents a detailed algorithm to compute the FSD.

A.6 Additional result on weight sign difference

Fig. 9 shows the weight sign difference of the adapter and normalizes it by the total number of adapter weights in BERT.

A.7 Additional results when mixing two adapters

Fig. 14 and 15 show model generalization when mixing two and multiple adapters across various tasks.

Table 5: Topic distribution on MRPC dataset

#Topic	MRPC
1	said, court, company, would, official, statement, decision, made, state, appeal, two, board
2	said, year, people, president, program, time, million, two, last, house, official, weapon
3	said, million, would, state, period, compared, men, get, democratic, plan, company, united, also, could
4	percent, share, cent, million, stock, point, nasdaq, billion, new, index, trading, rose, per, year
5	said, also, state, iraq, center, united, attack, hospital, killed, war, three, american, people
6	said, two, home, police, told, state, friday, last, year, federal, company, yesterday, national
7	standard, poor, index, chief, point, said, percent, justice, one, spx, broader, executive, three
8	said, analyst, expected, street, many, suit, call, yesterday, angeles, wall, los, research, one, change, according
9	case, said, court, filed, death, also, charged, lawsuit, charge, state, found, reported, office, cancer
10	said, would, server, window, network, one, new, microsoft, also, taken, people, company

Table 6: Topic distribution on QNLI dataset

#Topic	QNLI
1	city, american, south, large, west, season, de, roman, service, art, london, first, located, street, new
2	state, united, new, including, people, city, national, million, school, north, government, army, many, within, building
3	also, system, later, early, used, based, part, control, four, use, death, official, known, act, called
4	group, language, east, among, found, common, company, india, federal, movement, population, early, included, production, range
5	the, church, term, example, university, greek, german, like, english, specie, god, word, per, old, one
6	form, although, following, law, central, rule, culture, without, often, modern, territory, society, treaty, considered, christian
7	war, world, british, life, development, empire, first, region, community, year, france, though, time, set, began
8	well, three, include, place, power, party, league, may, needed, right, one, political, club, a, event
9	became, first, time, john, film, president, number, year, french, one, day, land, america, process, le
10	century, music, around, house, home, period, age, record, late, established, several, standard, time, world, river

Algorithm 4: Mixing Sparse Adapters with weight sign difference

Input: k domain-specific adapters, the FSD matrix $\mathbf{S}_{k\times k}$, l number of mix adapters.
Output: Average(sparse_candidates)

dense_candidates $\leftarrow \{\}$
Compute the average of the difference in weight
sign: $average_S = mean(S, axis = 1)$
Select the top l smallest adapters ($\mathbf{S}_l$) to mix based
on the average weight sign difference
dense_candidates $\leftarrow \mathbf{S}_l$
For each adapter, compute the average fraction of
weight sign different in each layer with corresponding layers from other adapters.
Get the top m layer with the highest fraction of
weight sign conflict to prune
sparse_candidates $\leftarrow$ Prune(dense_candidates)
return average(sparse_candidates)

A.8 Mixing Sparse Adapters with weight sign difference

Alg. 4 shows details of mixing sparse adapter with sign conflict information.

Table 7: Topic distribution on QQP dataset

#Topic	QQP
1	like, become, feel, get, job, movie, good, student, want, engineering, girl, website, sex, study, go
2	best, way, difference, learn, whats, money, make, online, book, india, buy, start, good, language, programming
3	much, best, time, weight, year, lose, old, place, month, day, iphone, read, possible, class
4	thing, day, business, get, first, going, example, one, prepare, video, woman, word, men
5	work, note, india, indian, ever, computer, black, r, science, you, help, rupee, different
6	would, life, trump, world, country, new, donald, war, india, win, happen, president, clinton, hillary
7	get, friend, used, long, why, bad, back, see, take, cant, good, facebook, system, relationship, person
8	someone, love, english, one, know, improve, account, people, get, instagram, tell, average, hair, password
9	mean, app, song, name, android, give, bank, right, what, company, india, working, get, now, create
10	people, quora, question, think, do, me, answer, google, stop, use, state, get, many, live

Table 8: Topic distribution on RTE dataset

#Topic	RTE
1	year, bank, world, ago, police, place, human, people, said, man, problem, game, many, took, explosion
2	people, attack, california, killed, life, united, day, lost, air, one, space, injured, national, capital, said
3	oil, said, nuclear, company, new, president, iran, million, military, john, un, country, bush, price
4	said, world, state, united, minister, country, million, people, nobel, south, peace, war, trade, prize, mexico
5	woman, corp, parliament, case, confirmed, said, rabies, represented, cause, poorly, fire, president, police, loss
6	year, new, said, one, would, died, university, show, company, family, first, service, since, country, home
7	state, iraq, said, bush, bomb, found, used, water, home, killed, caused, damage, one, police
8	party, police, president, new, two, officer, name, drug, state, prime, people, minister, last, year, democratic
9	new, said, government, year, iraq, would, york, official, today, baghdad, also, euro, announced, percent, minister
10	said, year, leader, new, sanfrancisco, work, justice, two, president, government, end, free, guerrilla

Table 9: Topic distribution on SST2 dataset

#Topic	SST2
1	film, really, enough, movie, something, make, interesting, many, like, subject, intelligent, laugh, short
2	movie, bad, film, better, great, fun, one, look, director, story, ultimately, smart, cinema, put
3	performance, funny, way, moment, film, cast, another, screen, yet, big, work, perfect, made
4	new, material, ve, movie, rather, film, special, seen, minute, enjoyable, might, offer, story, effect
5	comedy, drama, thriller, romantic, documentary, actor, moving, clever, funny, sometimes, pleasure, often, movie, film
6	work, film, movie, hard, well, keep, filmmaker, ever, life, original, sense, dull, quite, could
7	like, feel, movie, much, people, film, make, see, get, character, one, thing
8	good, real, film, worth, fascinating, make, time, lack, bit, amusing, humor, tale, pretty, run
9	character, one, best, film, movie, story, far, compelling, two, every, year, picture, little
10	love, audience, film, story, character, seems, entertainment, way, powerful, care, take, one, movie, spirit

Table 10: Topic distribution on RCT dataset

#Topic	RCT
1	group, patient, week, randomized, study, received, control, year, mg, randomly, placebo, day
2	patient, session, visit, cohort, failure, lesion, myocardial, hospital, twice,death, heart, infarction
3	analysis, using, data, model, used, test, sample, analyzed, regression, characteristic, time, collected, cell, method, performed
4	outcome, primary, month, patient, baseline, measure, score, secondary, treatment, scale, assessed, symptom, week, followup
5	risk, associated, level, weight, factor, effect, disease, body, increased, diabetes, insulin, high, glucose, change, activity
6	study, patient, treatment, effect, therapy, efficacy, may, effective, result, evaluate, safety, weather, clinical, outcome, intervention
7	group, difference, significant, significantly, compared, control, treatment, score, higher, lower, time, observed, rate
8	trial, study, randomized, intervention, care, health, controlled, clinical, quality, life, conducted, prospective, effectiveness, child, number
9	patient, event, surgery, adverse, postoperative, complication, procedure, pain, undergoing, surgical, rate, incidence, common, infection, injection
10	mean, respectively, ratio, patient, group, median, versus, interval, year, day, month

Table 11: Topic distribution on Tweets dataset

#Topic	Tweets
1	new, get, here, music, home, cool, playing, free, want, fun, season, shop, update, reason
2	day, one, night, time, good, week, last, never, first, get, year, got, lot, today
3	day, father, love, happy, time, weekend, take, friday, dad, fathersday, model
4	want, bull, up, do, help, trump, whatever, direct, dominate, waiting, libtard, yet, sleep, post
5	thankful, need, good, positive, orlando, morning, city, tear, news, blessed, friend, dream, bing, yeah, bong
6	user, amp, day, see, cant, go, like, new, today, one, people, get, wait, make
7	birthday, like, positive, affirmation, happy, baby, amp, god, girl, woman, feel, hate, hot, you
8	love, work, life, happy, happiness, make, always, food, quote, smile, wedding, moment, right, feeling, music
9	healthy, blog, gold, silver, altwaystoheal, forex, healing, grateful, dog, buffalo, peace, really, story
10	love, me, smile, summer, beautiful, fun, cute, girl, selfie, friend, sun, instagood, beach, photo

Table 12: Topic distribution on IMDB dataset

#Topic	IMDB
1	story, film, life, movie, character, one, love, time, people, see, way, family, would, well
2	movie, like, one, good, really, it, film, bad, see, even, time, would, make, get
3	get, one, man, the, go, woman, take, back, he, find, there, scene, two, girl
4	hamilton, gadget, arkin, scooby, talespin, stallion, smoothly, tenderness, shaggy, gil, inspector, keller, nevada, hopelessness
5	war, american, documentary, soldier, political, world, german, country, history, america, military, army, hitler
6	bollywood, indian, kapoor, khan, akshay, fi, amitabh, ramones, verhoeven, christina, sci, braveheart, kumar, chiller
7	film, one, the, scene, character, story, director, much, plot, well, even, work, time
8	film, role, performance, great, play, best, good, cast, one, actor, comedy, john
9	show, series, episode, year, tv, time, great, first, kid, dvd, one, funny, still, watch
10	match, matthau, luke, shakespeare, neil, bruce, scarface, boxing, hamlet, elvis, branagh, lucas, polanski

Table 13: Topic distribution on Ag News dataset

#Topic	Ag News
1	palestinian, said, iraqi, killed, iraq, reuters, attack, baghdad, arafat, israeli, bomb, scored, force, city
2	win, world, first, point, coach, cup, lead, victory, team, second, no, champion, night, final
3	president, afp, said, minister, election, bush, leader, india, state, reuters, prime, united
4	reuters, oil, price, stock, new, search, dollar, google, market, york, rate, apple, share, record
5	court, drug, say, ap, could, may, new, year, eu, case, said, state, scientist, trial
6	space, nasa, canadian, dec, press, former, nba, williams, winter, houston, monday, arsenal, sunday
7	said, company, inc, million, deal, corp, billion, sale, year, percent, reuters, buy, business
8	microsoft, new, software, internet, service, system, computer, technology, phone, ibm, music, online, web, company
9	china, police, said, reuters, people, worker, british, government, official, party, japan, group, chinese
10	game, new, year, red, one, time, season, first, team, series, last, york

Table 14: Topic distribution on Financial dataset

#Topic	Financial
1	company, finnish, new, plant, finland, construction, order, line, contract, service, unit, production, investment
2	company, share, bank, said, also, capital, start, issue, term, financial, price, business, executive, dividend
3	eur, profit, sale, net, operating, million, period, quarter, compared, loss, year
4	finnish, said, today, million, company, first, helsinki, year
5	company, mobile, said, phone, nokia, solution, business, pretax, finland, network, product, group, store, customer
6	market, board, option, company, share, stock, director, member, concerning, meeting, general, bank, flow, chairman
7	share, company, group, lower, helsinki, stock, president, capital, holding, new, right
8	service, finland, customer, corporation, company, electronics, solution, industry, business, helsinki, ltd, group
9	company, expected, sale, said, people, production, paper, year, finland, plant, cut, staff, expects
10	euro, service, company, item, nokia, excluding, technology, business, mobile, device, market, product

Table 15: Topic distribution on Authorship dataset

#Topic	Authorship
1	one, would, may, people, year, even, could, time, last, minister, public, police, many, blair, say
2	one, would, war, farmer, even, new, blair, bush, could, need, time, iraq, much, week
3	labour, new, people, government, tax, year, time, even, public, brown, blair, party, money
4	would, one, government, new, world, year, labour, much, state, blair, last, british
5	new, public, government, labour, people, year, one, would, may, way, time, make, right, life, need
6	people, time, public, said, even, government, lord, like, party, make, day
7	one, bush, american, world, year, right, war, child, people, british, state, new
8	people, one, child, like, time, family, get, year, burrell, may, still, even, much
9	would, one, blair, bush, war, nuclear, even, it, new, make, could, weapon, people, party
10	would, one, year, people, could, even, royal, like, woman, time, war, right, iraq

Table 16: Topic distribution on Wiki Toxic dataset

#Topic	Wiki Toxic
1	page, talk, edit, please, user, edits, wikipedia, editor, comment, block, blocked, editing, discussion, thanks, stop
2	image, use, you, copyright, page, fair, picture, please, medium, wikipedia, see, template, deleted, file, photo
3	article, deletion, deleted, page, please, tag, may, speedy, notable, talk, guideline, subject, wikipedia, criterion, add
4	nigger, hate, bitchfuck, faggot, lol, class, rape, fat, asshole, mama, fucker, hairy, ha, boymamas
5	like, know, get, people, it, think, you, want, one, time, go, thing, me, really
6	state, english, country, american, language, people, name, war, city, world, government, history, british, jew, group
7	fuck, ass, suck, fucking, shit, u, hi, cunt, school, moron, go, bitch, shut, cock, dick
8	utc, year, new, game, redirect, song, old
9	page, wikipedia, talk, help, please, link, welcome, question, article, thank, thanks, like, name, best
10	article, one, would, source, also, think, section, fact, see, it, like, point, say, time, reference

Task	Learning rate	epoch	batch size	warmup	weight decay	adapter size
$\text{BERT}_{\text{BASE}}$						
MNLI	4e-4	20	32	0.06	0.1	256
MRPC	4e-4	5	32	0.06	0.1	256
QNLI	4e-4	20	32	0.06	0.1	256
QQP	4e-4	20	32	0.06	0.1	256
RCT	4e-4	20	32	0.06	0.1	256
RTE	4e-4	5	32	0.06	0.1	256
SST2	4e-4	10	32	0.06	0.1	256
Tweets	4e-4	5	32	0.06	0.1	256
IMDB	4e-4	5	32	0.06	0.1	256
Ag News	4e-4	20	32	0.06	0.1	256
Financial	4e-4	5	32	0.06	0.1	256
Authorship	4e-4	5	32	0.06	0.1	256
$\text{RoBERTa}_{\text{LARGE}}$						
MNLI	3e-4	20	64	0.6	0.1	64
MRPC	3e-4	5	64	0.6	0.1	64
QNLI	3e-4	20	64	0.6	0.1	64
QQP	3e-4	20	64	0.6	0.1	64
RCT	3e-4	20	64	0.6	0.1	64
RTE	3e-4	5	64	0.6	0.1	64
SST2	3e-4	10	64	0.6	0.1	64
Tweets	3e-4	5	64	0.6	0.1	64
IMDB	3e-4	5	64	0.6	0.1	64
Ag News	3e-4	20	64	0.6	0.1	64
Financial	3e-4	5	64	0.6	0.1	64
Authorship	3e-4	5	64	0.6	0.1	64

Table 17: Hyperparameter configurations for various tasks.

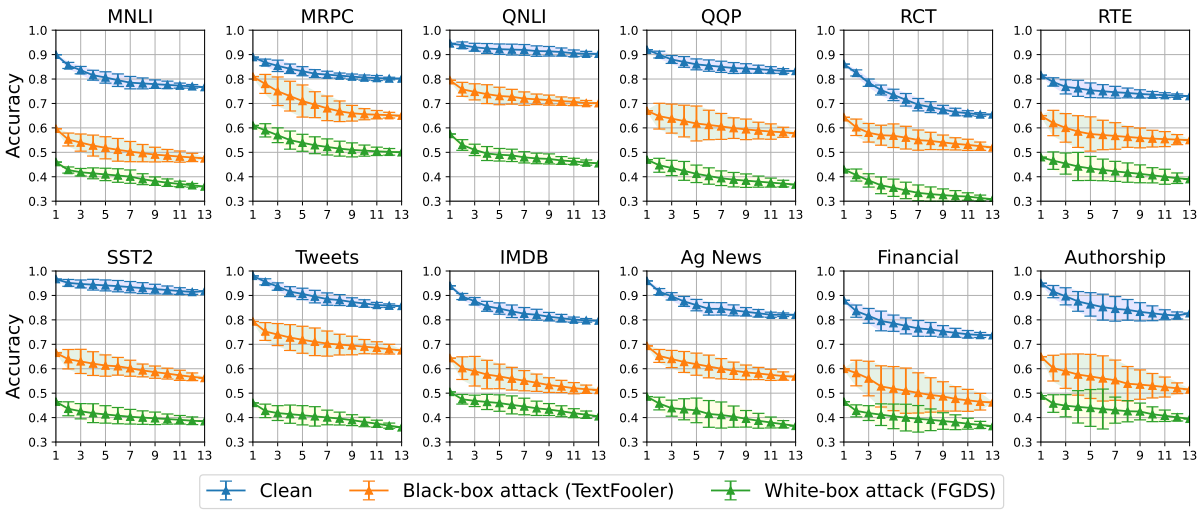

Figure 10: Accuracy of RoBERTa with Houlsby (Houlsby et al., 2019) across various distribution datasets. *The x-axis denotes the number of domain adapters to be mixed, ranging from 1 to 13.*

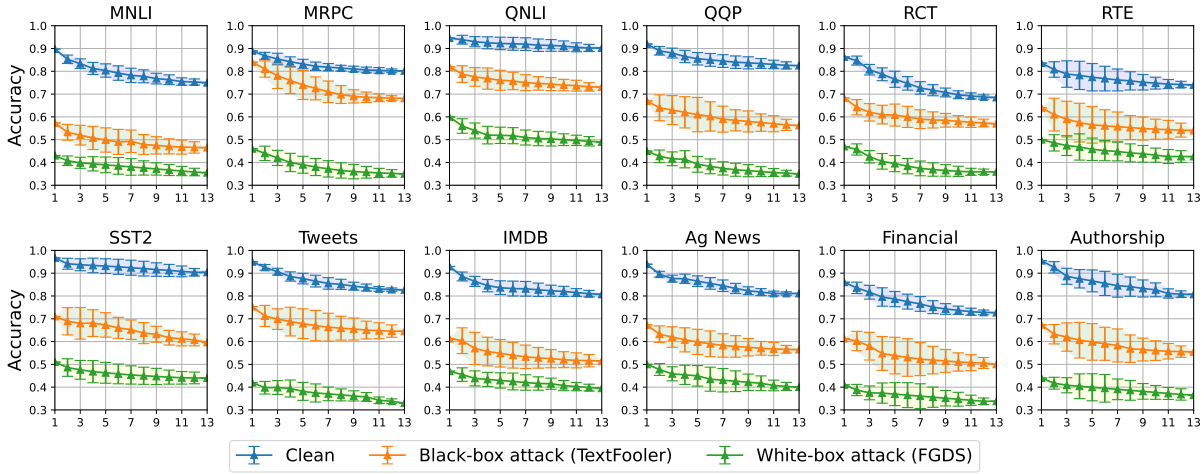

Figure 11: Performance Evaluation of RoBERTa Using the LoRA (Hu et al., 2022) across Varied Domain Datasets.

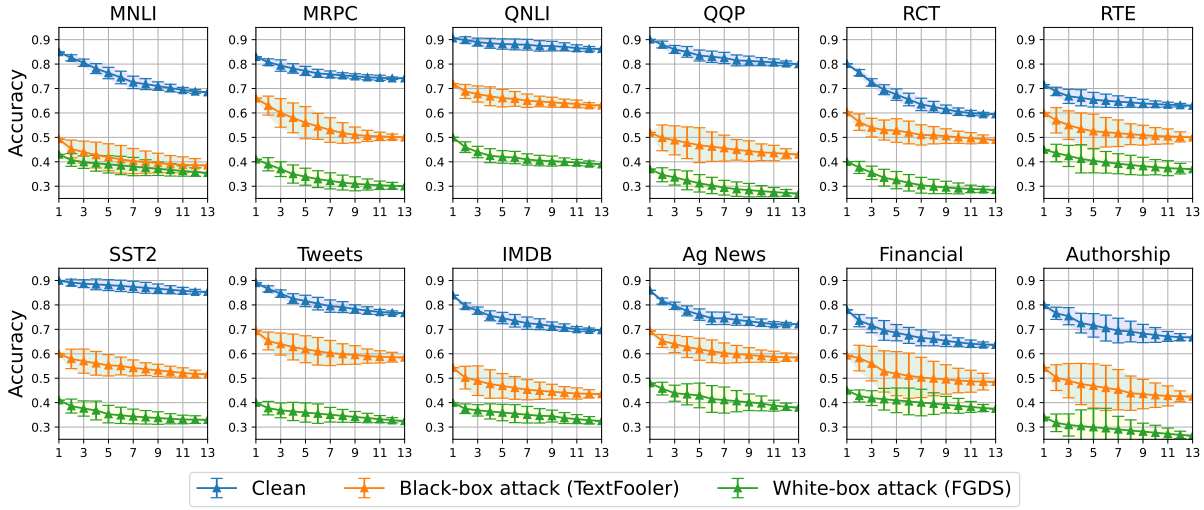

Figure 12: Performance Evaluation of BERT Using the Houlsby (Houlsby et al., 2019) across Varied Domain Datasets.

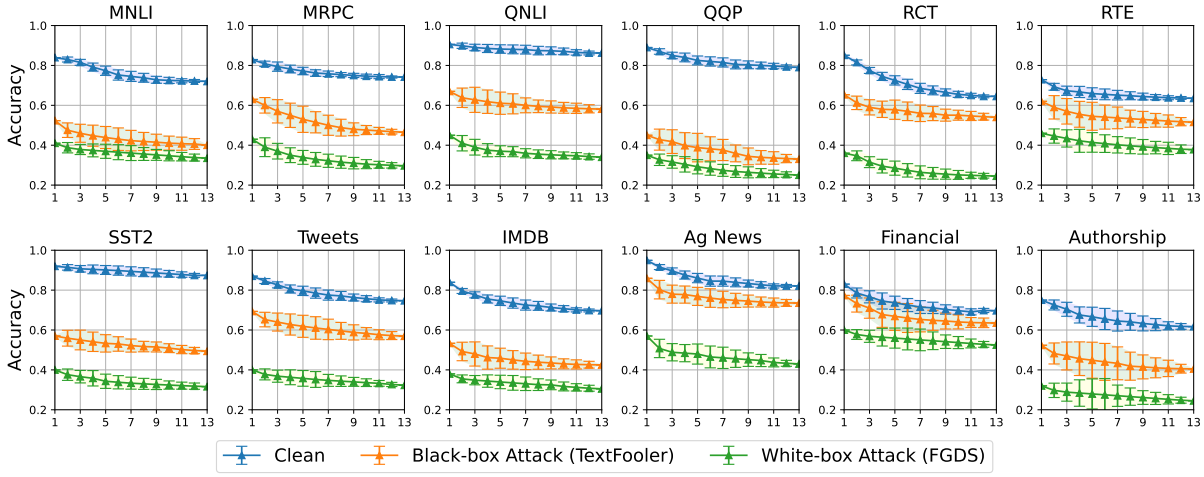

Figure 13: Performance Evaluation of BERT Using the Pfeiffer (Pfeiffer et al., 2021) across Varied Domain Datasets.

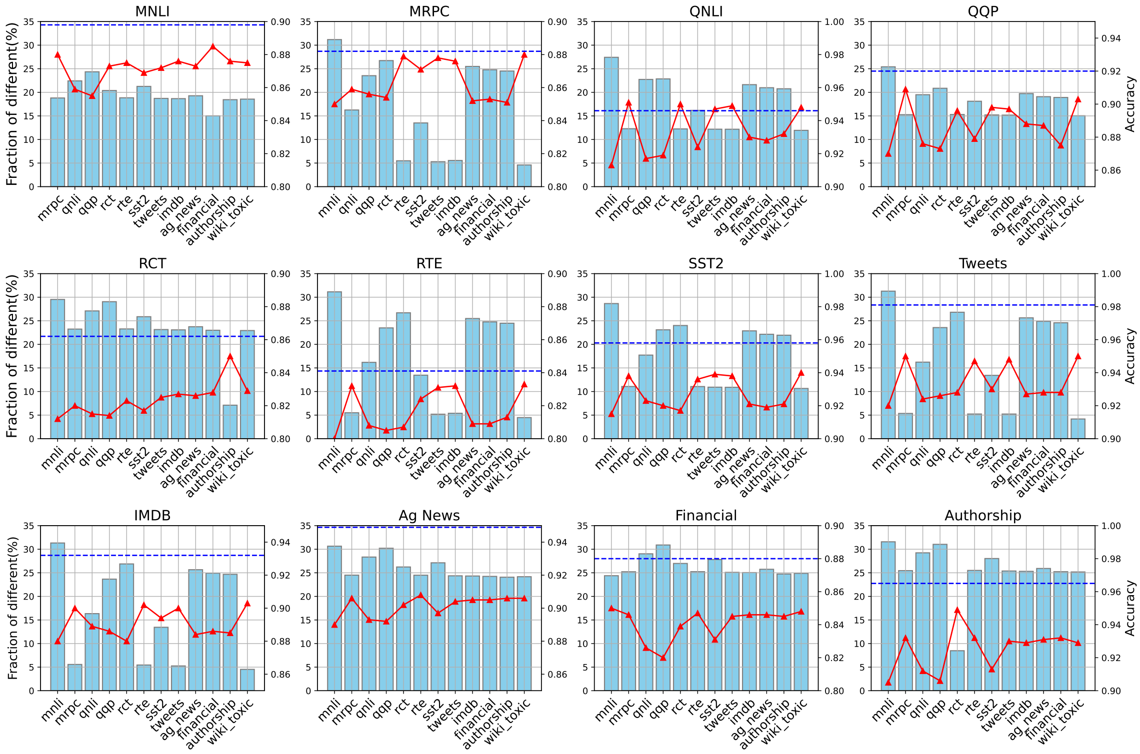

Figure 14: Fraction of weights altering direction during the consolidation of two adapters. The sky-blue bar represents the fraction of weight sign conflicts between two (k=2) adapters (left y-axis). The *dashed blue line* denotes the accuracy achieved by a standalone adapter trained on a specific task. While the *solid red line* illustrates the variations in accuracy when merging the adapter with another task's adapter.

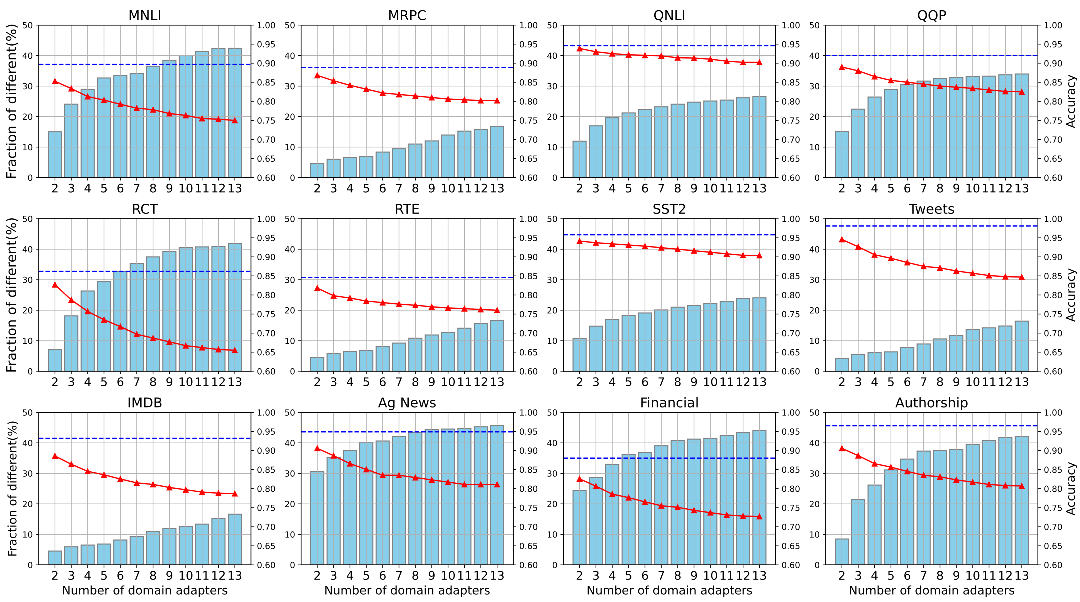

Figure 15: Fraction of weights changing direction during the mixing of multiple adapters, ranging from 2 to 13. The sky-blue bar represents the fraction of weight sign conflicts between k (from 1 to 13) adapters (left y-axis). The *dashed blue* line corresponds to the accuracy of a single adapter trained on a specific task, while the *solid red line* depicts the fluctuation in task accuracy resulting from merging the adapter with another task's adapter.