Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.057443

ARTICLE

AdaptForever: Elastic and Mutual Learning for Continuous NLP Task Mastery

Ke Chen[1,2], **Cheng Peng**[1,2], **Xinyang He**[1,2], **Jiakang Sun**[1,2], **Xu Liu**[1,2], **Xiaolin Qin**[1,2,*] **and Yong Zhong**[1,2,*]

[1]Chengdu Institute of Computer Applications, Chinese Academy of Sciences, Chengdu, 610213, China
[2]University of Chinese Academy of Sciences, Beijing, 101408, China

*Corresponding Authors: Xiaolin Qin. Email: qinxl2001@126.com; Yong Zhong. Email: zhongyong@casit.com.cn

Received: 17 August 2024; Accepted: 10 December 2024; Published: 06 March 2025

ABSTRACT: In natural language processing (NLP), managing multiple downstream tasks through fine-tuning pre-trained models often requires maintaining separate task-specific models, leading to practical inefficiencies. To address this challenge, we introduce AdaptForever, a novel approach that enables continuous mastery of NLP tasks through the integration of elastic and mutual learning strategies with a stochastic expert mechanism. Our method freezes the pre-trained model weights while incorporating adapters enhanced with mutual learning capabilities, facilitating effective knowledge transfer from previous tasks to new ones. By combining Elastic Weight Consolidation (EWC) for knowledge preservation with specialized regularization terms, AdaptForever successfully maintains performance on earlier tasks while acquiring new capabilities. Experimental results demonstrate that AdaptForever achieves superior performance across a continuous sequence of NLP tasks compared to existing parameter-efficient methods, while effectively preventing catastrophic forgetting and enabling positive knowledge transfer between tasks.

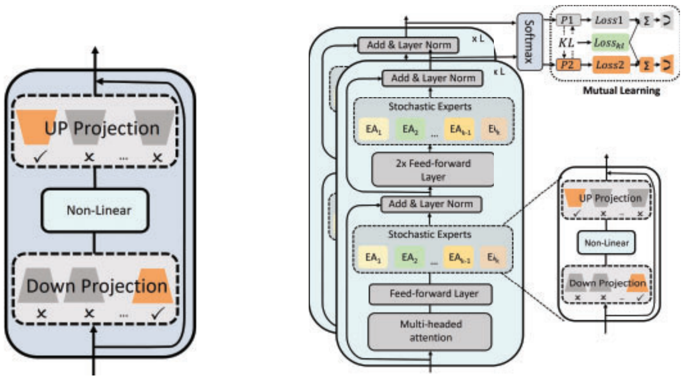
KEYWORDS: Adapter-tuning; large language model; pre-trained language model; parameter-efficient fine tuning; continue learning; mutual learning; mixture of expert

1 Introduction

In the rapidly evolving field of NLP, the refinement of large pre-trained language models stands at the forefront [1,2], driving advancements across a spectrum of applications. Traditionally, these models are fine-tuned for specific tasks, a process that, despite its effectiveness, is marred by significant computational demands and inefficiencies [3]. As exemplified by models like GPT-3 (Generative Pre-trained Transformer3) [4], which feature colossal parameter sizes, the fine-tuning approach is increasingly becoming unsustainable. It is hindered by extensive resource requirements and practical deployment challenges [5].

In response, researchers have pioneered innovative methods, such as adapter modules [6] and low-rank matrix adjustments [7], to streamline the fine-tuning process by modifying only a fraction of the model's parameters. Although these techniques enhance efficiency and expedite task adaptation, they often struggle to utilize knowledge from previous tasks effectively, an essential requirement for continual and multi-task learning [8,9]. Recent advancements, such as adapter modules [10] and other parameter-efficient fine-tuning approaches [11], have addressed some of the computational challenges, but they still face significant limitations in continual learning scenarios. In particular, when a model must switch between tasks or learn new ones, maintaining knowledge from previous tasks and preventing catastrophic forgetting remain substantial hurdles.

Inspired by the work [12], a parameter-efficient Mixture-of-Experts (MOE) [13] model that uses stochastic expert activation and consistency regularization for improved performance in machine translation tasks, our research introduces a novel stochastic expert adaptation mechanism, which decomposes the adapter module into multiple experts and accelerates training and inference speed through a random selection process, depicted in Fig. 1a. This mechanism innovatively combines mutual learning—a concept explored across various machine learning domains [14,15]—with parameter-efficient fine-tuning of language models.



(a) Stochastic expert adaption (b) Training process

Figure 1: (a) The stochastic expert module is strategically positioned behind the feed-forward layer. In this configuration, the hidden states are subjected to a down projection, where a single expert is selected at random. Following this, the states are processed through a non-linear layer and subsequently up-projected. This architectural choice allows the model to learn effectively through the expert adapter, enhancing its adaptability and performance across a range of tasks. Throughout the entire training cycle, a pair of down-projections and up-projections is randomly activated within the expert adapter modules. (b) The AdaptForever architecture's training process involves feeding input data through initial layers to extract features, which are then processed by feed-forward layers to learn complex patterns. Following this, multiple expert adapters are engaged to encapsulate and store knowledge from previous tasks, enhancing the model's adaptability. Optionally, a mutual learning component allows different parts of the model to learn from each other, further improving performance. This iterative process ensures the model continuously refines its ability to handle new tasks while retaining valuable insights from past experiences

AdaptForever employs stochastic expert selection to foster a model architecture in which various expert modules are activated randomly. This randomness ensures the maintenance of parameter efficiency without compromising the model's adaptability and learning capacity. Such dynamic expert activation helps prevent overreliance on specific parameters and promotes a flexible adaptation process across tasks. The integration of EWC [16], an algorithm that mitigates catastrophic forgetting in neural networks by protecting weights critical to previously learned tasks, allowing for effective continual learning, serves as a vital complement to this structure. It introduces a targeted regularization constraint that preserves key parameters essential for past task performance. This safeguard helps prevent the loss of crucial knowledge when learning new tasks, facilitating a seamless blend of knowledge retention and acquisition. Moreover, the mutual learning component allows expert modules to share and integrate knowledge, thereby enhancing the model's comprehension and utilization of previously acquired information.

Our key contributions are succinctly summarized as follows:

1. The development of AdaptForever, an advanced NLP framework that uniquely merges stochastic expert adaptation with elastic weight consolidation.

2. This approach redefines continuous learning, optimizing the balance between minimal parameter adjustments and maximum knowledge utilization, thereby surpassing conventional adapter-based methods.
3. By enhancing the adaptability and versatility of language models, AdaptForever proves to be highly effective in managing a diverse range of tasks.

2 Related Work

2.1 Fine-Tuning Large Pre-trained Language Models: Approaches and Limitations

Traditional Fine-Tuning Methods. The field of NLP has undergone a substantial transformation with the introduction of large pre-trained language models, such as BERT (Bidirectional Encoder Representation from Transformers) [1], RoBERTa (Robustly Optimized BERT Pre-training Approach) [2] and GPT3 [4]. Originating from the Transformer architecture [15], these models have been trained on vast text corpora, enabling them to capture complex linguistic patterns. Their effectiveness across various NLP tasks is attributed to this extensive training, which allows them to generate and understand nuanced language with remarkable accuracy.

Fine-tuning is a critical process in leveraging these large models for specific downstream tasks. It involves adjusting the pre-trained model's parameters, Θ, to adapt to the nuances of a particular task. This is typically achieved by training the model on a task-specific dataset, D_{task}, which modifies the parameters to Θ'. The fine-tuning process can be mathematically represented as:

$$\Theta' = \Theta - \eta \cdot \nabla_{\Theta}\mathcal{L}\left(P\left(y|x;\Theta\right), D_{\text{task}}\right) \tag{1}$$

where η is the learning rate and $\nabla_{\Theta}\mathcal{L}$ is the gradient of the loss function $\mathcal{L}$.

While they are effective, the process of fine-tuning these models for specific tasks presents significant challenges, particularly as model sizes increase [17]. The necessity to update all model parameters for each task leads to substantial computational and storage requirements [18]. This limitation is particularly acute in multi-task and continual learning scenarios, where each task requires a unique model adaptation.

Parameter-Efficient Fine-Tuning (PEFT). To mitigate the challenges of traditional fine-tuning, PEFT methods have been developed. PEFT methods aim to reduce the computational and storage overhead by updating only a small subset of the model's parameters while striving for performance comparable to full-parameter fine-tuning. Adapter-based methods, for example, introduce task-specific adjustments by inserting adapters at various points within the model. In our exploration, three unique adapter designs are scrutinized within the Transformer architecture, each uniquely contributing to efficient model performance while minimizing parameter increase.

The archetype, **Adapter^{h}**, conceived by [6], is a compact module adeptly positioned between the self-attention and multi-layer perceptron (MLP) segments. It consists of a down-projection $W_{\text{down}} \in \mathbb{R}^{d\times a}$, reducing dimensionality to a bottleneck size $\boldsymbol{a}$, and an up-projection $W_{\text{up}} \in \mathbb{R}^{a\times d}$, reinstating the original hidden state dimension $\boldsymbol{d}$. The adapter's operational mechanism is encapsulated by the equation:

$$h = \sigma\left(hW_{\text{down}}\right)W_{\text{up}} + h \tag{2}$$

where σ is indicative of a non-linear activation function, with $\boldsymbol{h}$ representing the resultant hidden state output from each feed-forward layer.

In our experimental comparison (Section 4.2), we also evaluate two additional adapter designs. The first, proposed by [19] and referred to as **Adapter^{d}**, optimizes efficiency by applying the adapter layer solely after the MLP module and LayerNorm. Another work, proposed by [8] and referred to as **Adapter$^{\text{F}}$**, leverages knowledge from multiple tasks by first learning task-specific parameters (adapters) in a knowledge extraction stage and then combining these adapters in a separate knowledge composition step.

Adapter modules, while adding a small number of trainable parameters per task, may not fully capture the complexity required for more sophisticated tasks. This can lead to a limitation in the model's ability to learn certain nuances, especially when the task is significantly different from the original pre-training objectives [20].

Similarly, LoRA ((Low-Rank Adaptation)) introduces trainable low-rank matrices into transformer layers to effectively approximate weight updates. Given a pre-trained weight matrix $W \in \mathbb{R}^{d\times k}$, LoRA represents its update with a low-rank decomposition: $W + \Delta W = W + W_{\text{down}} W_{\text{up}}$, where $W_{\text{down}} \in \mathbb{R}^{d\times r}$ and $W_{\text{up}} \in \mathbb{R}^{r\times k}$ are the tunable parameters and the rank $r \ll min\,(d, k)$.

LoRA may struggle with handling multiple tasks simultaneously due to its limited trainable parameters [21]. This can lead to issues with catastrophic forgetting, where the model loses knowledge of previous tasks when learning new ones.

Prompting methods have indeed emerged as an innovative approach for fine-tuning large language models. By strategically adding specific prompts to the model's input, these methods guide the model to perform specific tasks without the need to update all model parameters. This technique has shown promise in zero-shot or few-shot learning scenarios, where model performance can be significantly improved with minimal additional training data [22]. However, prompting methods are not without their challenges. One of the main drawbacks is the difficulty in optimizing the prompts themselves [23]. Crafting effective prompts requires careful consideration and can be a complex process, adding an extra layer of complexity to the fine-tuning process.

Other parameter-efficient tuning methods include PST [24], which aims to reduce the number of trainable parameters involved in computing the importance of weights during sparse-aware training in downstream tasks, and BitFit [25], which focuses on adjusting only the bias terms rather than the entire set of parameters. These methods share a common paradigm of maintaining the core architecture of the pre-trained model while introducing efficient mechanisms for task-specific adaptations. By updating only a subset of the model's parameters, they achieve a balance between adaptation flexibility and computational efficiency.

Specifically, while AdaptForever shares similarities with adapter-based methods, it goes further by introducing stochastic expert activation and a mutual learning mechanism, which enables continual learning without requiring extensive parameter updates. This distinguishes it from previous works that rely on static adapter configurations or low-rank approximations that may struggle with continual learning and catastrophic forgetting.

2.2 Mutual Learning

Mutual learning [14], as a paradigm where models collaboratively and concurrently learn from each other, has shown significant promise in enhancing model performance, particularly in multi-task environments. This is evident in applications such as dialogue disentanglement [15], cross-lingual tasks [26], and misinformation detection [27].

A prominent example is the THOR (Transformer with Stochastic Experts) [12], a model that employs a sparsely activated approach, leveraging the mutual learning mechanism for training. THOR demonstrates how mutual learning can obviate the need for traditional gating mechanisms, leading to more efficient learning processes.

The core concept of mutual learning involves updating model parameters not only based on individual task losses but also incorporating insights from peer models:

$$\Theta_i^{(t+1)} = \Theta_i^{(t)} - \eta\nabla\left(\mathcal{L}_i\left(\Theta_i\right) + \sum_{j\neq i}\alpha_{ij}KL\left(p_j \| p_i\right)\right) \tag{3}$$

Here, $\Theta_i^{(t)}$ denotes the parameters of model $\boldsymbol{i}$ at iteration $\boldsymbol{t}$, $\mathcal{L}_i$ is the loss function for the task associated with model $\boldsymbol{i}$, α_{ij} represents the mutual learning rate between models $\boldsymbol{i}$ and $\boldsymbol{j}$, and $KL\left(pj\|pi\right)$ means the Kullback-Leibler (KL) Divergence between the probability $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ given by model Θ_i and model Θ_j.

The application of mutual learning in diverse NLP tasks underscores its versatility and efficacy, paving the way for innovative approaches in handling complex language processing challenges.

2.3 Sparsely Activated Models and Mixture of Experts

Sparsely Activated Models (SAMs), including MoE architectures [13], have emerged as a scalable solution in neural networks, allowing for an expansion of parameters without a proportional increase in computational cost. Expanding on this concept, the work presented in [28] demonstrates an innovative approach to scaling neural machine translation Transformer models with sparsely gated mixture of experts to over 600 billion parameters. These models activate only a subset of experts for any given input, yet they have been criticized for their parameter inefficiency, whereas larger models do not necessarily lead to improved performance. Research [10] takes a critical look at the existing SAMs and introduces a novel concept of stochastic expert activation. This paper challenges the traditional gating-based routing methods, suggesting that they may not be more effective than a stochastic approach to routing inputs to experts.

Our proposed method distinguishes itself from the work that introduced stochastic expert activation in several key aspects. Architecturally, while the aforementioned work employed the feedforward layer as the MoE module for expert mapping, we have opted for a different strategy. We have integrated an adapter module into our model, using it as the expert for the selection process. This adapter module allows for more flexibility and adaptability in our model architecture.

Furthermore, there is a notable difference in the focus of our tasks. The previous work predominantly concentrated on translation tasks. In contrast, our research addresses a different area, filling a gap in language representation that was not covered in the initial exploration of stochastic expert activation.

2.4 Elastic Weight Consolidation

Elastic Weight Consolidation (EWC) [16,29,30] emerges as a pivotal technique in the landscape of continual learning, addressing the critical challenge of catastrophic forgetting.

The theoretical foundation of EWC is rooted in Bayesian statistics, where the penalty term is derived from the Fisher information matrix—a measure of the parameter importance concerning the model's prediction accuracy on prior tasks. This approach allows for a selective regularization process, concentrating the penalty on parameters with high relevance to the model's previously learned tasks, thereby mitigating the risk of catastrophic forgetting. The integration of EWC into neural network training algorithms has demonstrated remarkable efficacy in a variety of domains, including NLP [31], where models often encounter sequential learning scenarios. The regularization term added by EWC can be formulated as:

$$\mathcal{L}\left(\theta\right) = \mathcal{L}_{\text{new}}\left(\theta\right) + \sum_i \lambda\frac{F_i}{2}\left(\theta_i - \theta_{i,\text{old}}\right)^2 \tag{4}$$

where $L(\theta)$ is the total loss function, $L_{\text{new}}(\theta)$ is the loss for the new task, θ_i are the parameters of the model, $\theta_{i,\text{old}}$ are the parameter values after the previous task, F_i represents the Fisher information matrix that indicates the importance of parameters for the previous tasks, and λ is a hyper-parameter that controls the strength of the regularization.

3 Methodology

Recent advances in parameter-efficient training methods like LoRA [7] and Adapters [6] have predominantly focused on optimizing single tasks. We propose AdaptForever, an innovative framework that addresses these challenges by leveraging a mutual learning mechanism to guide expert selection and facilitate knowledge sharing between experts. Through this mechanism, AdaptForever randomly selects different expert modules to activate during each training step, enabling flexible adaptation across tasks. Additionally, we introduce Elastic Weight Consolidation (EWC) to preserve essential parameters for previously learned tasks, mitigating the risk of catastrophic forgetting and enabling effective continual learning. This design allows us to maintain computational efficiency while enhancing the model's adaptability and performance across multiple tasks. We utilize the RoBERTa model [2] and apply stochastic experts adapter with mutual learning strategies for continual learning, enhancing and extending current techniques. This approach achieves continual learning while maintaining parameter efficiency, offering both compactness and flexibility in handling diverse tasks.

3.1 Architectural Design

The AdaptForever model skillfully integrates stochastic expert selection with mutual learning within a transformer framework. It selects expert modules for each task in a stochastic manner, which is enhanced by the elastic weight consolidation technique. This technique introduces a critical regularization component that safeguards essential parameters, crucial for sustaining performance on previously mastered tasks. Thus, it ensures that the model's adaptability and capacity for knowledge transfer are maintained without compromising the retention of valuable prior knowledge.

Algorithm 1: The AdaptForever model

```
1: Input: Training datasets $\{D_1, D_2,\ldots, D_N\}$, initial dual model parameters $\{\theta_{(0)}, \theta'_{(0)}\}$, mutual learning rates $\{\alpha\}$, number of iterations T, Fisher in formation matrices $\{F_i\}$, EWC regularization strength $\{\lambda\}$
2: Output: Updated model parameters $\{\theta_{(N)}\}$
3: Initialization: Set $n = 0$. Initialize parameters $\theta_{(0)}, \theta'_{(0)}$ for dual model.
4: for each dataset iteration i from 1 to $N$ do
5:     Compute the task-specific loss $L(\theta_i)$ using $D_i$
6:     Compute the KL divergence $\frac{1}{2}(KL(p||p') + KL(p'||p))$
7:     if $n \geq 1$ then
8:         Compute the EWC regularization $\sum_i \frac{\lambda_i}{2} F_i(\theta_i - \theta_{i,old})^2$
9:     end if
10:    Update parameters of model i using gradient descent:
11:    $\theta_{i+1} = \theta_i - \nabla(L(\theta_i) + \alpha R_{consistency}(\theta, \theta') + \lambda L_{EWC})$
12:    $\theta'_{i+1} = \theta'_i - \nabla(L(\theta'_i) + \alpha R_{consistency}(\theta', \theta) + \lambda L_{EWC})$
13: end for
14: Return updated parameters, the better performance $\{\theta_{(N)}\}$.
```

Additionally, a consistency regularization mechanism (see Section 3.2) optimizes expert engagement based on the learning context and performance metrics, further complementing this approach. Collectively, these mechanisms significantly bolster the model's adaptability across a continuous range of tasks. The entire process is methodically detailed in Algorithm 1, providing a clear roadmap for the model's operation.

3.1.1 Stochastic Expert Adapter Modules

At the core of the AdaptForever model lie our innovative adaptive expert modules. Unlike traditional adapters that typically feature a single down and up projection, our adapters are equipped with multiple down and up projections. This unique design significantly expands the adapter modules' capacity, enabling them to handle more complex and varied language processing tasks.

During both the training and inference phases, our model randomly activates a single pair of these down and up projections as shown in Fig. 1a. This stochastic activation method helps maintain a manageable computational load, even as the adapters offer increased capacity. Moreover, the architecture's multiple projections per module provide the model with a wider array of options and enhanced adaptability, making it well-suited for a diverse range of tasks.

3.1.2 Mutual Learning Mechanism

AdaptForever incorporates a crucial architecture, the mutual learning mechanism. In this setup, expert modules enhance their capabilities by learning from both task-specific data and shared knowledge, thereby fostering a comprehensive understanding and holistic adaptation across tasks. This collaborative learning strategy not only supports dynamic adaptation but also helps in preventing overfitting. Fig. 1b provides a visual representation of the training process, illustrating the sequential steps and interactions within the AdaptForever model's learning strategy.

Within the framework of our model, the training process meticulously balances task-specific optimizations with the enforcement of output consistency across modules. This dual focus is encapsulated in the training objective as follows:

$$\mathcal{L}_{\text{AdpF}} = \mathcal{L}\left(X, Y; \theta\right) + \lambda \cdot \mathcal{R}_{\text{consistency}}\left(\theta\right) \tag{5}$$

where

- X and Y represent the input and label sets for all tasks, respectively.
- θ denotes the model's parameters.
- $\mathcal{L}$ is the loss function utilized for the tasks.
- $\mathcal{R}_{\text{consistency}}$ is the consistency regularization term, applied to maintain uniformity in module outputs.
- λ is a hyper-parameter that adjusts the intensity of the consistency regularization.

The consistency regularization term, $\mathcal{R}_{\text{consistency}}$, leverages the Kullback-Leibler divergence to assess and minimize differences between the output probability distributions of various expert modules:

$$\mathcal{R}_{\text{consistency}}\left(\theta\right) = \frac{1}{2}\left(\text{KL}\left(P\left(\cdot|X;\theta_i\right)||P\left(\cdot|X;\theta_j\right)\right) + \text{KL}\left(P\left(\cdot|X;\theta_j\right)||P\left(\cdot|X;\theta_i\right)\right)\right) \tag{6}$$

3.1.3 Enhanced Training Regimen with EWC

The AdaptForever framework, augmented with elastic weight consolidation, meticulously addresses the dual imperatives of continual learning: task-specific specialization and broad generalization. EWC enhances the framework by introducing a regularization term, specifically designed to preserve crucial parameters that affect performance on previously learned tasks. Consequently, the total loss function when tackling a new task is formulated as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{AdpF}} + \mathcal{L}_{\text{EWC}} \tag{7}$$

where $\mathcal{L}_{\text{AdpF}}$ denotes the original loss function of Eq. (5), encompassing the consistency regularization. $\mathcal{L}_{\text{EWC}}$ represents the elastic weight consolidation regularization term, given by:

$$\mathcal{L}_{\text{EWC}} = \sum_i \frac{\lambda_i}{2} F_i \left(\theta_i - \theta_{i,\text{old}}\right)^2 \tag{8}$$

Here, θ_i are the current model parameters, $\theta_{i,\text{old}}$ are the parameters after training on previous tasks, and $\boldsymbol{F_i}$ is the Fisher information matrix that quantifies the importance of each parameter to the tasks learned thus far. The hyper-parameter λ_i control the strength of the regularization, striking a balance between knowledge retention and acquisition.

3.2 Analysis of Continual Training and Consistency Regularization

The training regimen within the AdaptForever framework is meticulously crafted to cater to the intrinsic requirements of continual learning. It operates on a dual-faceted approach that not only strives for task-specific optimization but also emphasizes the enforcement of consistency across the various expert modules. This is achieved through an astute consistency regularization mechanism integral to the model. For demonstration and derivation purposes, we limit our proof to only two datasets $\boldsymbol{D_A}$ and $\boldsymbol{D_B}$, which are assumed to be independent of each other within this context.

Given a parameter set θ and data $\boldsymbol{D}$, where $\boldsymbol{D}$ is divided into two independent parts $\boldsymbol{D_A}$ and $\boldsymbol{D_B}$, the posterior probability $p(\theta|D)$ can be expressed as:

$$p(\theta|D) = \frac{p(\theta, D)}{p(D)} = \frac{p(\theta, D_A, D_B)}{p(D_A, D_B)} = \frac{p(\theta, D_B|D_A)}{p(D_B|D_A)} = \frac{p(D_B|\theta, D_A)\, p(\theta|D_A)}{p(D_B|D_A)} = \frac{p(D_B|\theta)\, p(\theta|D_A)}{p(D_B)} \tag{9}$$

Assuming independence between $\boldsymbol{D_A}$ and $\boldsymbol{D_B}$, the equation simplifies to the last line. Thus, the log posterior is:

$$\log p(\theta|D) = \log p(D_B|\theta) + \log p(\theta|D_A) - \log p(D_B) \tag{10}$$

This relationship shows that the log posterior can be decomposed into a log-likelihood term involving $\boldsymbol{D_B}$, a log prior term involving $\boldsymbol{D_A}$, and a normalizing constant term involving $\boldsymbol{D_B}$.

For the three terms on the right side, the first term, $\log p(D_B|\theta)$, is the negative loss function on $\boldsymbol{D_B}$, denoted by $-L_B(\theta)$. The third term, $\log p(D_B)$, is a parameter-independent constant. Thus, the optimization objective becomes (maximum a posteriori estimation):

$$\theta = \arg\max \log p(\theta|D) = \arg\max\left(-L_B(\theta) + \log p(\theta|D_A)\right) = \arg\min L_B(\theta) - \log p(\theta|D_A) \tag{11}$$

This objective effectively minimizes the loss on $\boldsymbol{D_B}$ while maximizing the parameter posterior on $\boldsymbol{D_A}$, thereby avoiding catastrophic forgetting.

What we need to find is $p(\theta|D_A)$, that this posterior is intractable. Therefore, we use the Laplace approximation method, following the paper [17].

As seen in comparison to Maximum likelihood estimation (MLE), the Eq. (11) essentially includes an additional term $-\log p(\theta|D_A)$. Thus, this term can be interpreted as a regularization component within the structured risk framework.

The goal of the Laplace approximation is to approximate the posterior $p(\theta|D_A)$ with a Gaussian distribution. Assuming it is a Gaussian distribution, we express the posterior $p(\theta|D_A)$ as:

$$p(\theta|D_A) = \mathcal{N}(\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma}}\exp\left(-\frac{(\theta-\mu)^2}{2\sigma^2}\right) \tag{12}$$

Thus, the logarithm of the posterior $\log p(\theta|D_A)$ is:

$$\log p(\theta|D_A) = \log\left(\frac{1}{\sqrt{2\pi\sigma}}\right) - \frac{(\theta-\mu)^2}{2\sigma^2} \tag{13}$$

Let $f(\theta) = \log p(\theta|D_A)$, and suppose $\theta = \theta_A^*$ yields the optimum. At $\theta = \theta_A^*$, the Taylor expansion $f(\theta)$ is given by:

$$f(\theta) = f(\theta_A^*) + \left.\frac{\partial f(\theta)}{\partial\theta}\right|_{\theta=\theta_A^*}(\theta-\theta_A^*) + \frac{1}{2}(\theta-\theta_A^*)^T\left.\frac{\partial^2 f(\theta)}{\partial\theta^2}\right|_{\theta=\theta_A^*}(\theta-\theta_A^*) + O(\theta-\theta_A^*) \tag{14}$$

Since θ^* maximizes $f(\theta)$ (here it appears to be proven that such a maximum exists), the first derivative of $f(\theta)$ at $\theta = \theta^*$ is zero, and the second derivative is negative (indicating a negative definite Hessian matrix). Ignoring terms of third order and higher, $f(\theta)$ can be approximated as:

$$f(\theta) = \log\left(\frac{1}{\sqrt{2\pi\sigma}}\right) - \frac{(\theta-\mu)^2}{2\sigma^2} \approx f(\theta_A^*) + \frac{1}{2}(\theta-\theta_A^*)^2 f''(\theta_A^*) \tag{15}$$

For convenience, we will not emphasize the matrix form, remember that $f''(\theta^*)$ is a Hessian matrix. Since $\log\left(\frac{1}{\sqrt{2\pi\sigma}}\right)$ and $f(\theta_A^*)$ are constants independent of the parameters, we have:

$$-\frac{(\theta-\mu)^2}{2\sigma^2} = \frac{(\theta-\theta_A^*)^2}{2}f''(\theta_A^*) \tag{16}$$

Hence:

$$\mu = \theta_A^*, \quad \sigma^2 = -\frac{1}{f''(\theta_A^*)} \tag{17}$$

The two hyper-parameters of the Gaussian distribution are thus determined, and the posterior $p(\theta|D_A)$ is obtained. Now, the optimization objective can be written as:

$$\theta = \arg\min L_B(\theta) - \log p(\theta|D_A) = \arg\min L_B(\theta) - \frac{1}{2}(\theta-\theta^*)^2 f''(\theta_A^*) \tag{18}$$

Since $f(\theta_A)$ is a constant, it can be omitted. Therefore, maximizing $p(\theta|D_A)$ becomes equivalent to maximizing $\frac{1}{2}(\theta-\theta_A)^2 f''(\theta_A^*)$. Now, the problem turns to how to compute $f''(\theta^*)$. Although $f''(\theta^*)$ can be directly calculated, the parameter θ is an n-dimensional vector, meaning that $f''(\theta^*)$ is an $n\times n$ Hessian matrix, which leads to significant computational cost in terms of both time and space. To reduce this computational overhead, we transform the Hessian matrix into the Fisher information matrix. The final loss function is given by:

$$L(\theta) = L_B(\theta) - \frac{\lambda}{2} f''(\theta_A^*)(\theta - \theta_A^*)^2 = L_B(\theta) + \frac{\lambda}{2}\sum F_i(\theta_i - \theta_A^*)^2 \tag{19}$$

This formulates the loss function that incorporates a regularization term with the Fisher information matrix to penalize deviations from the optimal parameters found after training on task A.

The consistency regularization employs a Kullback-Leibler divergence-based regularize that quantitatively measures and then minimizes the variance between the predictions of various expert modules. By incorporating this regularize, AdaptForever adeptly maintains a balance between the model's capacity to specialize in individual tasks and its ability to generalize across them. It mitigates the potential overfitting that might arise from the stochastic nature of expert module activation and fosters a collaborative learning paradigm.

Moreover, this approach is instrumental in reinforcing the transferability of learned representations, a quality that is paramount in multi-task learning. It propels each expert module to not just learn from the data specific to its task but also to draw from the collective insights accrued across different tasks. As a result, the model exhibits a robust performance, characterized by its dexterity in handling diverse linguistic challenges.

In essence, the training regimen and consistency regularization employed by AdaptForever are critical for realizing a balanced and integrated learning outcome. Fig. 2 visually contrasts various regularization strategies, highlighting their distinct impacts on parameter adjustment within a continual learning context.

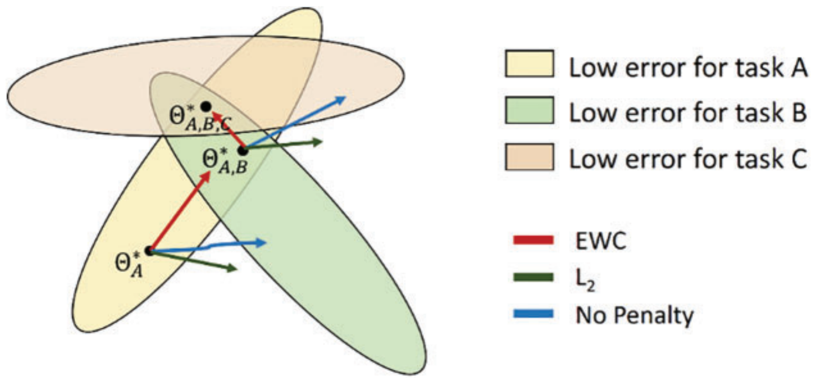


Figure 2: This image demonstrates the parameter evolution in a multi-task learning scenario. Each shaded region represents a zone of low error for a corresponding task. The red arrow (EWC) signifies an approach that adjusts parameters from one task to another while considering the importance of each parameter to prevent catastrophic forgetting of the original task

4 Experiments

4.1 Datasets and Experimental Setup

Tasks and Datasets. Our experiments were conducted on the GLUE benchmark, a collection of diverse NLP tasks designed to evaluate model performance across a range of language understanding challenges. GLUE's comprehensive suite includes tasks like textual entailment (MNLI), sentiment analysis (SST-2), and question answering (QNLI), among others. The variety of tasks allowed us to rigorously assess AdaptForever's capabilities in multi-task learning, with performance evaluated using standard metrics such as accuracy and F1 score for respective tasks.

Model Configuration and Implementation Details. We utilized two versions of the RoBERTa model [2] for our study. For the base model, we set the batch size to 64 and performed a hyper-parameter search over learning rates and epochs within the GLUE dataset, with rates ranging from $\{2e{-}5, 5e{-}4\}$ and epochs between $\{3, 20\}$. The choice of batch size was based on memory constraints and the efficiency of model training. A smaller batch size allows the model to learn more granularly from the data but can increase training time. We balanced these factors based on the model size and available resources. For the large model, a batch size of 32 was used, along with a similar hyper-parameter search for learning rates in the range of $\{1e{-}5, 2e{-}4\}$. The selection of learning rates and the number of epochs was guided by the outcomes of hyper-parameter searches, aiming to identify the optimal model performance. We drew on experimental results from [6] to determine a reasonable range for these parameters. Our models were implemented using the PyTorch framework, with data preprocessing steps including tokenization and normalization applied before training. To ensure fair comparisons, we report the average results over multiple training runs for each task.

4.2 Baselines Performance Comparison

In this section, we meticulously selected three baseline model methods for comparative analysis. For each method, we assigned two different adapter sizes. For the RoBERTa-base model, we chose 64 as the adapter size; for the RoBERTa-large model, we utilized 128 as the adapter size. As for the **Adapter^d** model, we selected the first two-thirds of its layers for investigation, and the choice was guided by experimental results from the original Adapter^d framework [19]. All experimental results were averaged after being tested three times to ensure the accuracy and reliability of the data.

Table 1 highlights the notable efficiency of AdaptForever in multi-task learning contexts. When compared to baseline models, AdaptForever exhibits consistently superior performance, showcasing a remarkable capacity for rapid adaptation across diverse tasks.

Table 1: For the evaluation metrics in the GLUE benchmark, the table reports the overall (matched and mismatched) accuracy for MNLI, Matthew's correlation for CoLA, Pearson correlation for STS-B, and accuracy for other tasks. All metrics follow the principle that higher scores indicate better performance. The performance of the RoBERTa models, both base and large versions, across various adapter tuning methods, including AdaptForever, is showcased

Model & Method	**Train. Param.**	**MNLI**	**SST-2**	**MRPC**	**CoLA**	**QNLI**	**QQP**	**RTE**	**SST-B**	**Avg.**
*RoB$_{base}$	125.0 M	87.6	94.8	90.2	63.6	92.8	**91.9**	78.7	**91.2**	**86.4**
*Adap^d	0.8 M	87.1 ± 0.0	94.2 ± 0.1	88.5 ± 1.1	60.8 ± 0.4	**93.1** ± 0.1	90.2 ± 0.0	71.5 ± 2.7	89.7 ± 0.3	84.4
*Adap^h	0.9 M	87.3 ± 0.1	94.7 ± 0.3	88.4 ± 0.1	62.6 ± 0.9	93.0 ± 0.2	90.6 ± 0.0	75.9 ± 2.2	90.3 ± 0.1	85.4
*Adap^p	0.9 M	87.5 ± 0.2	94.6 ± 0.1	88.6 ± 0.6	62.7 ± 1.2	**93.2** ± 0.2	90.5 ± 0.3	76.7 ± 1.8	89.9 ± 0.1	85.5
Adapt Forever	1.0 M	**87.8** ± 0.2	**95.0** ± 0.1	88.7 ± 0.1	**64.3** ± 0.5	92.6 ± 0.3	91.1 ± 0.1	**81.6** ± 1.2	90.1 ± 0.2	**86.4**
*RoB$_{large}$	355.0 M	90.2	96.4	**90.9**	68.0	94.7	**92.2**	86.6	**92.4**	88.9
*Adap^d	2.4 M	**90.5** ± 0.3	**96.6** ± 0.2	89.7 ± 1.2	67.8 ± 2.5	**94.8** ± 0.3	91.7 ± 0.2	80.1 ± 2.9	91.9 ± 0.4	87.9
*Adap^h	3.0 M	90.2 ± 0.3	96.1 ± 0.3	90.2 ± 0.7	68.3 ± 1.0	**94.8** ± 0.2	91.9 ± 0.1	83.8 ± 2.9	92.1 ± 0.1	88.4
*Adap^p	3.0 M	89.9 ± 0.5	96.2 ± 0.3	88.7 ± 2.9	66.5 ± 1.2	94.7 ± 0.2	92.1 ± 0.1	83.4 ± 1.1	91.0 ± 1.7	87.8
AdaptForever	3.0 M	90.1 ± 0.1	**96.6** ± 0.0	88.7 ± 0.2	**69.9** ± 0.8	94.5 ± 0.3	92.0 ± 0.3	**88.1** ± 0.9	92.1 ± 0.5	**89.0**

Note: (1) Results marked with an asterisk * have been sourced from previously published works. (2) Avg. means the model's average performance on GLUE benchmark. (3) Bold number represents the best performance on one dataset.

With the RoBERTa-base configuration, the AdaptForever method utilizes merely 1 M additional parameters yet achieves an average performance on par with the fine-tuning approach. This is especially evident in tasks such as RTE and CoLA, where AdaptForever either outperforms or equals the performance of the base model. Such results underscore the efficacy of AdaptForever's approach to parameter efficiency, suggesting that it does not just preserve, but may amplify, the model's generalization prowess across various tasks. When employing the RoBERTa-large model, AdaptForever maintains robust performance, with particular success noted in the SST-2 and CoLA tasks. This indicates its effective application in conjunction with more intricate models and extensive datasets.

To conclusively establish the model's superiority, we conducted comparative evaluations across several bottleneck sizes against models using an adapter architecture. The results, illustrated in Fig. 3, reveal that our model's comprehensive performance outstrips that of the competing models. This comparative study bolsters the assertion that AdaptForever excels in parameter efficiency while retaining or enhancing generalization capabilities across a spectrum of tasks.

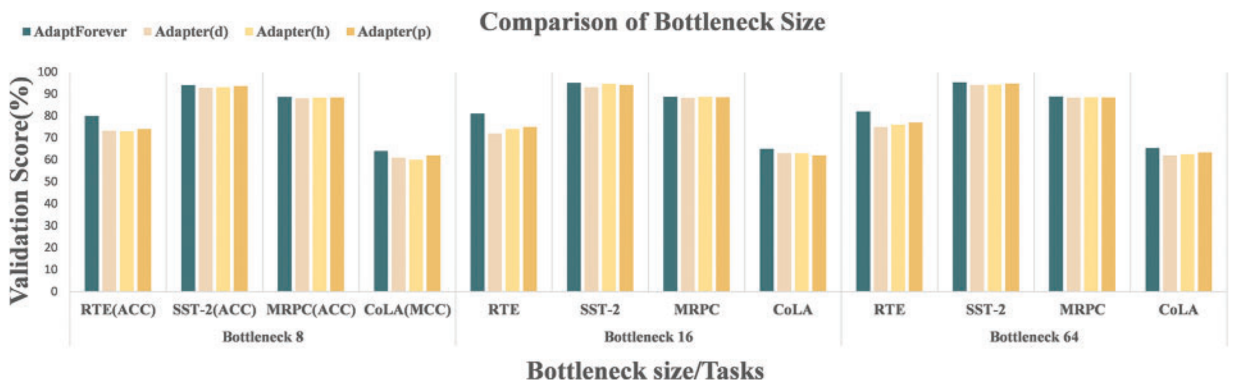


Figure 3: The performance of four different models (AdaptForever, Adapter^d, Adapter^h, and Adapter^p) across four datasets (RTE, SST-2, MRPC, and CoLA) with varying bottleneck sizes (8, 16, and 64). Each dataset is represented by three sets of bars, each set corresponding to a model's validation score at a specific bottleneck size. The validation scores are presented as percentages, with different colors indicating different models

4.3 Model Analysis

4.3.1 Model Convergence

Fig. 4 depicts the convergence patterns of the AdaptForever model, examining the interplay between the number of experts (N) and bottleneck sizes (D) during training. For N = 4, a larger bottleneck size of D = 64 leads to a notably quicker and higher climb in accuracy, suggesting that increased bottleneck capacity can accelerate learning. Conversely, with N = 4 and N = 8, accuracy levels across varying bottleneck sizes tend to converge, indicating that a greater number of experts might offset the influence of bottleneck size on the rate of convergence. This trend points to the possibility that the model's learning efficiency could be optimized by carefully calibrating the number of experts against the chosen bottleneck size.

4.3.2 Cross-Dataset Performance

The heatmaps in Fig. 5 present the continual learning transfer performance of the AdaptForever method employing the RoBERTa-base model across different tasks. The performance is measured relative to the original performance and displayed as percentages. In tasks such as RTE, a larger bottleneck size (D = 64) generally correlates with higher performance, suggesting that more capacity in the bottleneck can improve transfer learning in tasks requiring complex reasoning.

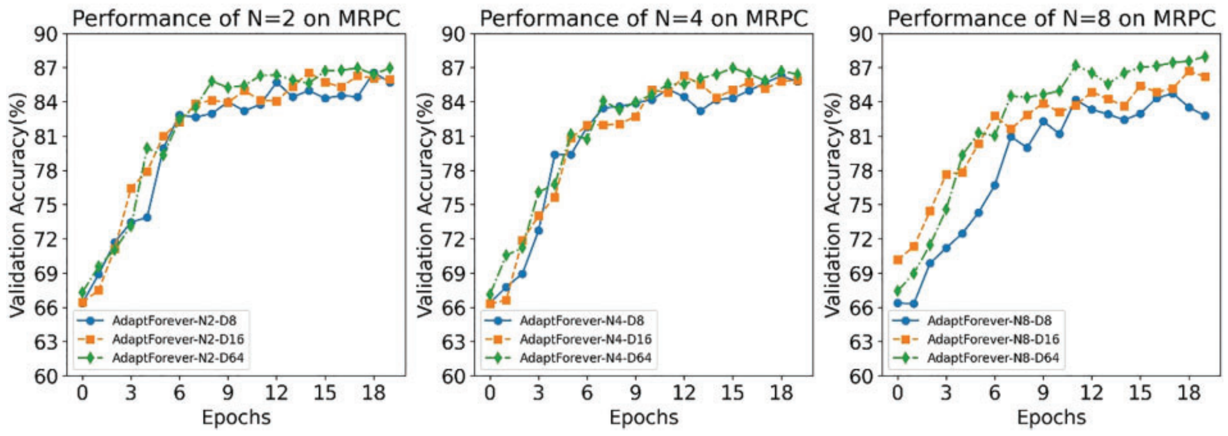


Figure 4: This image presents three parallel line charts, each displaying the performance of different numbers of experts (N) on the MRPC dataset. The titles represent models with 2, 4, and 8 experts. In each chart, there are three lines corresponding to different bottleneck sizes (D): D = 8, D = 16, and D = 64. Each line traces the variation in validation accuracy from 0 to 20 epochs

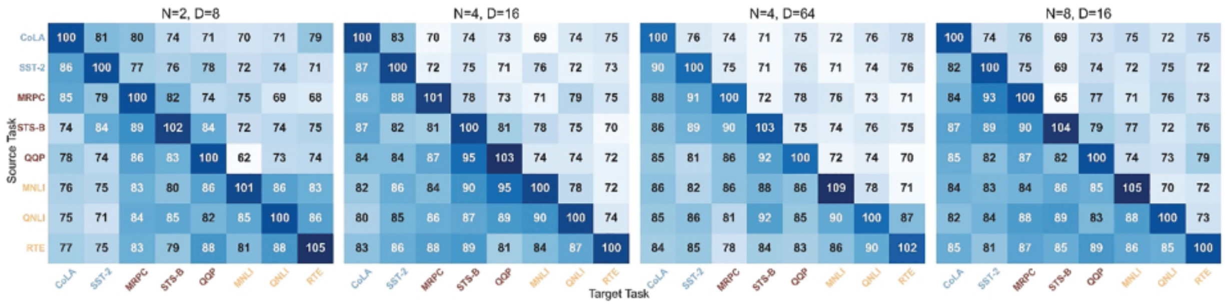


Figure 5: This heatmap showcases the continual learning performance of the AdaptForever methods using the RoBERTa-base model, highlighting the effect of varying bottleneck sizes (D) and the number of experts (N) on the learning dynamics. The chart illustrates two key aspects of the model's performance: the lower triangular matrix shows the impact of learning new tasks on the performance of previously learned tasks, while the upper triangular matrix indicates the model's zero-shot transfer performance on new, unlearned tasks. Color coding for task names differentiates the task types: Blue signifies single-sentence tasks, Brown represents similarity and paraphrase tasks, and Yellow corresponds to inference tasks. The percentages displayed represent the ratio of transfer learning performance to original performance, allowing for an at-a-glance comparison of the model's versatility and learning efficiency across various datasets and task types

For MRPC and SST-B, performance remains consistently high across all configurations, indicating that AdaptForever is robust in tasks involving sentiment analysis, regardless of bottleneck size or number of experts. CoLA and SST-2 show a more mixed response, with some configurations like N = 2, and D = 8.

4.4 Balancing Expert Adapter Quantity and Size

Our investigation into the AdaptForever model's structure entailed a thorough examination of not only the number of expert adapters but also the delicate interplay between their quantity and size. While integrating expert adapter modules within every layer, we found, as evidenced by Table 2 (RTE), that increasing the number of expert adapters does not linearly improve performance. In particular, Table 2 (CoLA) shows that models equipped with up to 4 expert adapters exhibit significant gains in efficiency across most tasks. However, the benefits plateau when expanding beyond this number, indicating a saturation point where additional adapters lead to diminishing performance improvements. The SST-2 results as shown in Table 2 (SST-2) demonstrate robust and consistent performance across all configurations, with accuracy scores above 94% and optimal performance (95.07%) achieved at both D = 16/N = 8 and D = 64/N = 4, indicating that moderate bottleneck sizes coupled with an appropriate number of expert adapters can effectively maximize model performance on sentiment analysis tasks.

Table 2: The tables compare the effects of varying bottleneck sizes (D) and numbers of expert adapters (N) on model performance

N	D											
	RTE			SST-2			MRPC			CoLA		
	8	16	64	8	16	64	8	16	64	8	16	64
2	77.98	79.06	80.10	94.83	94.95	95.02	85.90	84.87	86.46	58.31	58.29	62.24
4	74.72	78.34	79.49	94.83	94.95	**95.07**	86.70	86.27	86.88	58.81	60.38	63.36
8	75.09	78.73	**81.63**	94.95	**95.07**	94.96	86.94	86.64	**88.69**	58.82	59.36	**64.32**

Note: The best results were highlighted in bold.

Moreover, the size of the expert adapters emerges as a crucial factor. Our investigation into various bottleneck sizes (8, 16, and 64) revealed that larger sizes are generally associated with enhanced performance. Table 2 (MRPC) demonstrates that these performance gains level off at a bottleneck size of 64, suggesting an optimal threshold. Expanding the bottleneck size beyond this point does not yield commensurate improvements, particularly when considering increased computational costs.

The AdaptForever model's performance across tasks like RTE, SST-2, MRPC, and CoLA is influenced by the number and size of expert adapters, with up to 4 adapters showing significant efficiency gains. Larger bottleneck sizes, up to 64, generally enhance performance but with diminishing returns beyond this point. The model achieves optimal performance with a balanced configuration of adapters, emphasizing the importance of a careful calibration approach to maximize gains while optimizing computational resources.

5 Conclusions

This paper introduces an advanced continual learning strategy, showcasing the substantial benefits of integrating stochastic expert selection with a mutual learning paradigm. Our methodology enhances both the robustness and efficiency of continual learning. The AdaptForever approach effectively leverages the expansive parameter spaces inherent in large language models. It is designed to adeptly accumulate and apply knowledge from an array of completed tasks, facilitating a high degree of adaptability. Critically, it achieves this versatility without sacrificing the computational efficiency essential for practical applications.

The method's reliance on extensive parameter spaces may lead to challenges in optimization, especially when dealing with limited data scenarios or when the model's complexity exceeds the capacity of the available computational resources. Additionally, the approach's effectiveness might be contingent upon the quality and diversity of the tasks it is exposed to, which could potentially limit its applicability in more homogeneous or constrained learning environments.

As for future work, there are several promising directions. First, it would be valuable to explore methods that can enhance the AdaptForever approach's optimization process, particularly in scenarios with limited data. This could involve developing more efficient initialization strategies or incorporating techniques that reduce the risk of overfitting. Second, researching how the AdaptForever approach can be adapted to handle tasks with varying degrees of relatedness could broaden its applicability. This might involve exploring mechanisms that allow the model to selectively attend to different expert networks based on task characteristics.

Acknowledgement: The authors would like to thank the University of Chinese Academy of Sciences and Chengdu Information Technology of Chinese Academy of Sciences Co., Ltd., for their support and resources. We extend our gratitude to our peers and research partners whose thoughtful discussions and feedback enriched this review.

Funding Statement: This research was partly supported by the National Key R&D Program of China (No. 2023YFB3308601), Sichuan Science and Technology Program (2024NSFJQ0035, 2024NSFSC0004), and the Talents by Sichuan provincial Party Committee Organization Department.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Ke Chen, Cheng Peng, Yong Zhong; data collection: Xu Liu; analysis and interpretation of results: Jiakang Sun, Xinyang He; draft manuscript preparation: Ke Chen, Xiaolin Qin. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are openly available on GitHub at https://github.com/nyu-mll/GLUE-baselines (accessed on 09 December 2024).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Devlin J, Chang M-W, Lee K, Toutanova K. BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; 2019; Minneapolis, MN, USA. Vol. 1, p. 4171–86. doi:10.18653/v1/N19-1423.
2. Zhuang L, Wayne L, Ya S, Jun Z. A robustly optimized BERT pre-training approach with post-training. In: Proceedings of the 20th Chinese National Conference on Computational Linguistics; 2021; Huhhot, China: Chinese Information Processing Society of China. p. 1218–27.
3. Strubell E, Ganesh A, McCallum A. Energy and policy considerations for deep learning in NLP. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics; 2019; Florence, Italy. p. 3645–50. doi:10.18653/v1/P19-1355.
4. Brown TB, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, et al. Language models are few-shot learners. Paper presented at: Proceedings of the 34th International Conference on Neural Information Processing Systems; 2020; Vancouver, BC, Canada.
5. He J, Zhou C, Ma X, Berg-Kirkpatrick T, Neubig G. Towards a unified view of parameter-efficient transfer learning. arXiv:2110.04366. 2021.
6. Houlsby N, Giurgiu A, Jastrzebski S, Morrone B, de Laroussilhe Q, Gesmundo A, et al. Parameter-efficient transfer learning for NLP. arXiv:1902.00751. 2019.
7. Hu EJ, Shen Y, Wallis P, Allen-Zhu Z, Li Y, Wang S, et al. Lora: low-rank adaptation of large language models. arXiv:2106.09685. 2021.

8. Pfeiffer J, Kamath A, Cho RCK, Gurevych I. AdapterFusion: non-destructive task composition for transfer learning. In: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics; 2021. p. 487–503. doi:10.18653/v1/2021.eacl-main.39.
9. Stickland AC, Murray I. BERT and PALs: projected attention layers for efficient adaptation in multi-task learning. Paper presented at: Proceedings of the 36th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research; 2019.
10. Wang Y, Agarwal S, Mukherjee S, Liu X, Gao J, Awadallah HA, et al. AdaMix: mixture-of-adaptations for parameter-efficient model tuning. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing; 2022; Abu Dhabi, United Arab Emirates. p. 5744–60. doi:10.18653/v1/2022.emnlp-main.388.
11. Zhang Q, Chen M, Bukharin A, Karampatziakis N, He P, Cheng Y, et al. Adaptive budget allocation for parameter-efficient fine-tuning. In: International Conference on Learning Representations; 2023; Ithaca, New York.
12. Zuo S, Liu X, Jiao J, Kim YJ, Hassan H, Zhang R, et al. Taming sparsely activated transformer with stochastic experts. arXiv:2110.04260. 2021.
13. Shazeer N, Mirhoseini A, Maziarz K, Davis A, Le Q, Hinton G, et al. Outrageously large neural networks: the sparsely-gated mixture-of-experts layer. In: International Conference on Learning Representations; 2016; Ithaca, New York.
14. Zhang Y, Xiang T, Hospedales TM, Lu H. Deep mutual learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018; Salt Lake City, UT, USA: IEEE. p. 4320–8.
15. Jiang Z, Shi L, Chen C, Mu F, Zhang Y, Wang Q. MuiDial: improving dialogue disentanglement with intent-based mutual learning. In: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22); 2022; San Francisco, CA, USA. p. 4164–70.
16. Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, et al. Overcoming catastrophic forgetting in neural networks. Proc Nat Acad Sci. 2017;114(13):3521–6. doi:10.1073/pnas.1611835114.
17. Vaswani A. Attention is all you need. Advances in neural information processing systems. In: NIPS.17: Proceedings of the 31st International Conference on Neural Information Processing Systems; 2017; Red Hook, NY, USA: Curran Associates Inc. p. 6000–10.
18. Fedus W, Zoph B, Shazeer N. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. J Mach Learn Res. 2022;23(1):120.
19. Rücklé A, Geigle G, Glockner M, Beck T, Pfeiffer J, Reimers N, et al. AdapterDrop: on the efficiency of adapters in transformers. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing; 2021; Punta Cana, Dominican Republic. p. 7930–46. doi:10.18653/v1/2021.emnlp-main.
20. He R, Liu L, Ye H, Tan Q, Ding B, Cheng L, et al. On the effectiveness of adapter-based tuning for pretrained language model adaptation. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing; 2021; Stroudsburg, PA, USA. p. 2208–22. doi:10.18653/v1/2021.acl-long.172.
21. Li D, Ma Y, Wang N, Ye Z, Cheng Z, Tang Y, et al. MixLoRA: enhancing large language models fine-tuning with lora based mixture of experts. arXiv:2404.15159. 2024.
22. Schulhoff S, Ilie M, Balepur N, Kahadze K, Liu A, Si C, et al. The prompt report: a systematic survey of prompting techniques. arXiv:2406.06608. 2024.
23. Kepel D, Valogianni K. Autonomous prompt engineering in large language models. arXiv:2407.11000. 2024.
24. Li Y, Luo F, Tan C, Wang M, Huang S, Li S, et al. Parameter-efficient sparsity for large language models fine-tuning. arXiv:2205.11005. 2022.
25. Ben Zake E, Goldberg Y, Ravfogel S. BitFit: simple parameter-efficient fine-tuning for transformer-based masked language-models. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics; 2022; Dublin, Ireland. p. 1–9. doi:10.18653/v1/2022.acl-short.1.
26. Conneau A, Khandelwal K, Goyal N, Chaudhary V, Wenzek G, Guzmán F, et al. Unsupervised cross-lingual representation learning at scale. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics; 2020; Stroudsburg, PA, USA. p. 8440–51. doi:10.18653/v1/2020.acl-main.747.

27. Kumari R, Ashok N, Ghosal T, Ekbal A. Misinformation detection using multitask learning with mutual learning for novelty detection and emotion recognition. Inf Process Manage. 2021;58(5):102631. doi:10.1016/j.ipm.2021.102631.
28. Lepikhin D, Lee H, Xu Y, Chen D, Firat O, Huang Y, et al. GShard: scaling giant models with conditional computation and automatic sharding. arXiv:2006.16668. 2020.
29. Huszár F. Note on the quadratic penalties in elastic weight consolidation. Proc National Acad Sci. 2018;115(11):E2496–7. doi:10.1073/pnas.1717042115.
30. Batra H, Clark R. EVCL: elastic variational continual learning with weight consolidation. arXiv:2406.15972. 2024.
31. Biesialska M, Biesialska K, Costa-jussC MR. Continual lifelong learning in natural language processing: a survey. In: Proceedings of the 28th International Conference on Computational Linguistics; 2020; Barcelona, Spain. p. 6523–41. doi:10.18653/v1/2020.coling-main.574.