Parameter Importance is Not Static: Evolving Parameter Isolation for Supervised Fine-Tuning

Zekai Lin[1,⋆], Chao Xue[5,⋆], Di Liang[1,2,†], Xingsheng Han[1], Peiyang Liu[3], Xianjie Wu[1], Lei Jiang[1], Yu Lu[1], Bob Simons[1,2], Shuang Liang[4], Minlong Peng[1,†]

[1] Tencent Hunyuan [2] Tencent Yuanbao [3] Peking University
[4] UESTC, China [5] University of New South Wales
{rcjgdds,xuechao8071}@gmail.com; {liangd17,mlpeng16}@fudan.edu.cn

Abstract

Supervised Fine-Tuning (SFT) of large language models often suffers from task interference and catastrophic forgetting. Recent approaches alleviate this issue by isolating task-critical parameters during training. However, these methods represent a static solution to a dynamic problem, assuming that parameter importance remains fixed once identified. In this work, we empirically demonstrate that parameter importance exhibits temporal drift over the course of training. To address this, we propose Evolving Parameter Isolation (EPI), a fine-tuning framework that adapts isolation decisions based on online estimates of parameter importance. Instead of freezing a fixed subset of parameters, EPI periodically updates isolation masks using gradient-based signals, enabling the model to protect emerging task-critical parameters while releasing outdated ones to recover plasticity. Experiments on diverse multi-task benchmarks demonstrate that EPI consistently reduces interference and forgetting compared to static isolation and standard fine-tuning, while improving overall generalization. Our analysis highlights the necessity of synchronizing isolation mechanisms with the evolving dynamics of learning diverse abilities.

1 Introduction

Supervised Fine-Tuning (SFT) has become the dominant paradigm for activating the specialized capabilities of Pre-trained Large Language Models (LLMs), aligning them with diverse applications ranging from complex logical reasoning to open-ended creative generation. While this paradigm has proven effective in single-task scenarios, adapting a single backbone to a heterogeneous task stream presents a fundamental conflict. Since all tasks share the same high-dimensional parameter space, the aggressive updates required to learn new capabilities often conflict with the representations of previously acquired knowledge. This parameter-level contention inevitably triggers the "seesaw phenomenon" (Yu et al., 2020) and catastrophic forgetting (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017), where improvements in one domain (e.g., coding) inadvertently degrade performance in others (e.g., mathematics or dialogue).

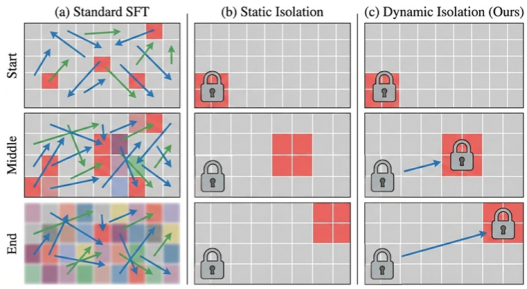

Figure 1: **Conceptual illustration of parameter isolation strategies.** (a) Standard SFT: parameters are updated globally, leading to interference. Arrows denote task-specific gradient directions. (b) Static Isolation: The pre-computed mask fails to cover critical parameters (red) as their importance distribution shifts during training. (c) Dynamic Isolation (Ours): The protection mask is dynamically updated to follow the trajectory of parameter importance.

To resolve this dilemma, prior research has explored parameter management strategies, building on the premise that different tasks rely on overlapping but distinct subsets of parameter space. This observation has motivated the development of Parameter-Efficient Fine-Tuning (PEFT) methods. Techniques such as Adapter modules (Houlsby et al., 2019) and Low-Rank Adaptation (LoRA) (Hu et al., 2022) introduce distinct, modular subspaces to compartmentalize task-specific knowl-

[1] ⋆ Equal Contribution.† Corresponding Author.
[2] This work was completed by Zekai Lin and Chao Xue under Di Liang's supervision.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 32846–32864

edge. More recently, a more direct approach known as Parameter Isolation has emerged as a promising direction. Methods in this category (Wang et al., 2025; Sorrenti et al., 2023) explicitly identify task-critical parameters within the original backbone and lock them to prevent destructive interference. Despite their empirical success, these isolation approaches rely on a key but largely unexamined assumption: they presume that the importance of parameters is a static property that, once identified at initialization, remains fixed throughout training.

We argue that this static assumption is misaligned with the highly dynamic nature of SFT. Fine-tuning is inherently an unstable process. As training progresses, the model's internal representations evolve, optimization trajectories shift, and the functional roles of specific parameters change significantly. We refer to this phenomenon as Parameter Importance Drift. For instance, parameters responsible for learning basic output formats in the early stages may become redundant later, while those governing complex reasoning logic may only become critical in later training stages. As visually illustrated in Figure 1, the set of task-critical parameters may drift over training; we verify this quantitatively later through mask-dynamics analyses. Consequently, static isolation mechanisms inevitably fall out of sync with the learning trajectory. This misalignment leads to two detrimental consequences: on one hand, the model wastes capacity by protecting obsolete parameters that no longer require safeguarding; on the other hand, it fails to lock emerging critical regions, leaving them vulnerable to being overwritten by subsequent updates.

To bridge this gap, we propose Evolving Parameter Isolation (EPI), a dynamic fine-tuning framework designed to synchronize protection mechanisms with the model's evolving optimization trajectory. Unlike prior works that rely on precomputed, static masks, EPI introduces an online importance estimation mechanism that continuously monitors gradient-based signals to track parameter sensitivity and update the isolation mask in real time. Because instantaneous gradients are noisy and gradient scales differ across layers, EPI combines temporal smoothing (EMA) with layer-wise normalization before selecting the protected subset. Using this evolving feedback, EPI periodically updates its isolation decisions, effectively creating a "moving shield" that adaptively locks currently critical parameters while releasing redundant ones for flexible adaptation. This mechanism resolves the conflict between Stability (retaining established knowledge) and Plasticity (absorbing new capabilities) by keeping the isolation strategy aligned with the learning trajectory.

Our contributions are summarized as follows: (1) We identify and quantify the temporal drift in parameter importance during the SFT process, providing empirical evidence that challenges the static assumption in parameter isolation research. (2) We propose EPI, a parameter-efficient framework that leverages online gradient statistics to dynamically update isolation masks without introducing extra trainable parameters. (3) Extensive experiments across diverse benchmarks covering reasoning, coding, and dialogue demonstrate that EPI consistently outperforms both standard SFT and static isolation baselines, yielding a more stable and generalizable adaptation across diverse tasks.

2 Methodology

In this section, we present Evolving Parameter Isolation (EPI) (illustrated in Figure 2), a framework explicitly designed to address the dynamic nature of parameter importance during the SFT of LLMs. We first provide a theoretical analysis of gradient interference via the lens of loss landscape geometry. Then, we detail the EPI framework, covering online importance estimation, adaptive layer-wise normalization, and the dynamic masking mechanism.

2.1 Problem Formulation and Gradient Interference

We consider the SFT of a pre-trained Large Language Model (LLM), parameterized by $\boldsymbol{\theta} \in \mathbb{R}^d$, on a heterogeneous stream of tasks $\mathcal{T}$. The global optimization objective is to minimize the expected risk across all tasks:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{k=1}^{K} \mathbb{E}_{\mathcal{D}_k} \left[\ell\big(f_{\boldsymbol{\theta}}(x), y\big) \right]. \quad (1)$$

However, simply minimizing this aggregate loss is insufficient due to parameter contention. In this multi-task setting, let $\boldsymbol{g}_i$ and $\boldsymbol{g}_j$ denote the gradients derived from two distinct tasks $\mathcal{T}_i$ and $\mathcal{T}_j$. We formally define Task Interference based on the cosine similarity between their update directions:

$$\mathcal{C}(\boldsymbol{g}_i, \boldsymbol{g}_j) = \frac{\boldsymbol{g}_i \cdot \boldsymbol{g}_j}{\|\boldsymbol{g}_i\| \|\boldsymbol{g}_j\|}. \quad (2)$$

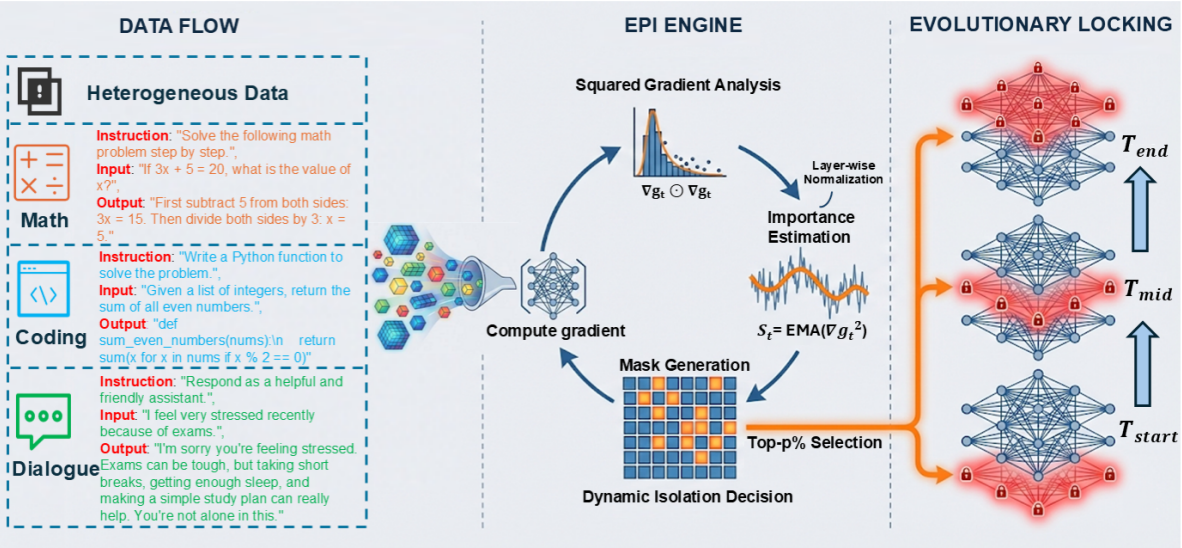

Figure 2: **Overview of the Evolving Parameter Isolation (EPI) framework.** The model trains on heterogeneous data (Left) while the EPI Engine (Middle) continuously estimates parameter sensitivity using exponential moving averages of squared gradients. These signals drive the Evolutionary Locking mechanism (Right), which dynamically updates isolation masks from T_{start} to T_{end} to protect shifting task-critical parameters.

When $\mathcal{C}(\boldsymbol{g}_i, \boldsymbol{g}_j) < 0$, the gradients are conflicting, leading to the "seesaw phenomenon" where reducing the loss for $\mathcal{T}_i$ increases the loss for $\mathcal{T}_j$. Our goal is to identify a dynamic parameter subspace where such conflicts are minimized.

2.2 Theoretical Motivation: Minimizing Forgetting

While Eq. (2) provides a macroscopic view of task interference, it does not localize which parameters are most responsible for such conflict. Instead, our goal is to derive a scalable signal that identifies which parameters should be protected.

To prevent catastrophic forgetting, we must ensure that the loss on previously learned capabilities $\mathcal{L}_{\text{old}}$ does not increase significantly. Assuming the model has converged to a local optimum $\boldsymbol{\theta}^*$ for previous tasks, we can approximate the loss increase caused by a parameter shift $\Delta\boldsymbol{\theta}$ using a second-order Taylor expansion (Kirkpatrick et al., 2017):

$$\mathcal{L}_{\text{old}}(\boldsymbol{\theta}^* + \Delta\boldsymbol{\theta}) \approx \mathcal{L}_{\text{old}}(\boldsymbol{\theta}^*) + \Delta\boldsymbol{\theta}^\top \nabla\mathcal{L}_{\text{old}}(\boldsymbol{\theta}^*) + \frac{1}{2}\Delta\boldsymbol{\theta}^\top \mathbf{H}\Delta\boldsymbol{\theta}, \quad (3)$$

where $\mathbf{H}$ is the Hessian matrix. Given the assumption of local optimality for previous tasks $\nabla\mathcal{L}_{\text{old}}(\boldsymbol{\theta}^*) \approx \mathbf{0}$, minimizing forgetting is equivalent to minimizing the quadratic term $\frac{1}{2}\Delta\boldsymbol{\theta}^\top \mathbf{H}\Delta\boldsymbol{\theta}$. This implies that parameters with high curvature (large Hessian values) are critical for preserving knowledge.

Based on this principle, traditional parameter isolation methods identify critical regions at initialization and fix the protection mask $\boldsymbol{m}_t$ throughout training (i.e., $\boldsymbol{m}_t \equiv \boldsymbol{m}_0$). However, we argue that parameter importance is not invariant but drifts over time. To quantify this effect, we define the Temporal Drift as the Hamming distance between ideal masks at different steps:

$$d_H(\boldsymbol{m}^*_{t_1}, \boldsymbol{m}^*_{t_2}) = \sum_{j=1}^{d} \mathbb{I}\left(m^{(j)}_{t_1} \neq m^{(j)}_{t_2}\right). \quad (4)$$

Our analysis (see Appendix) shows that d_H is significant, suggesting that static masks inevitably fall out of sync with the learning trajectory and necessitating a dynamic strategy.

2.3 Evolving Parameter Isolation (EPI)

To bridge the gap between static protection and dynamic learning, EPI introduces a time-varying isolation mechanism.

2.3.1 Online Importance Estimation via Fisher Approximation

Computing the exact Hessian $\mathbf{H}$ involves high computational and memory costs ($O(d^2)$) for LLMs. To circumvent this bottleneck, we leverage the theoretical insight that the Fisher Information Matrix (FIM) asymptotically approaches the Hessian near local minima (Pascanu and Bengio, 2013; Martens,

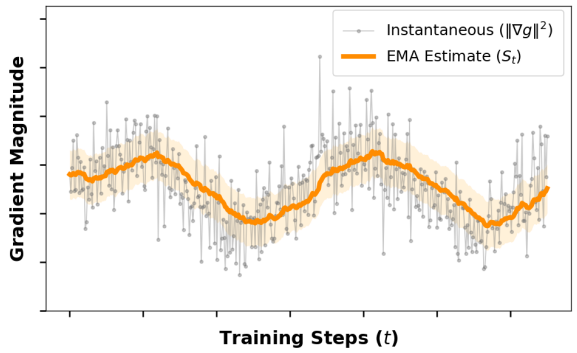

Figure 3: **Illustration of temporal smoothing.** The Instantaneous Gradient (Gray) exhibits high variance. In contrast, the EMA Estimate (Orange) effectively filters high-frequency noise, revealing the underlying importance trend ($\boldsymbol{S}_t$).

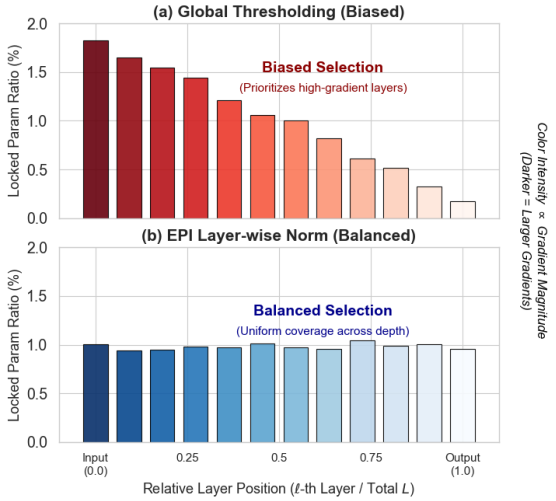

Figure 4: **Illustration of necessity of Layer-wise Normalization.** (a) Without normalization, global ranking causes severe bias, concentrating protection only on layers with high gradient norms (Red bars). (b) Our Adaptive Layer-wise Normalization rectifies this by projecting scores to a local scale, resulting in a uniform distribution of critical parameters (Blue bars) regardless of the layer's position.

2020). To ensure scalability, we adopt the Empirical Fisher approximation, restricting our estimation to the diagonal elements using squared gradients.

Let $\boldsymbol{g}_t = \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_t)$. We maintain a running sensitivity vector $\boldsymbol{S}_t \in \mathbb{R}^d$ via Exponential Moving Average (EMA):

$$\boldsymbol{S}_t = \beta \boldsymbol{S}_{t-1} + (1-\beta)(\boldsymbol{g}_t \odot \boldsymbol{g}_t), \tag{5}$$

where $\beta \in [0, 1)$ is a smoothing coefficient.

As illustrated in Figure 3, the instantaneous squared gradients ($\boldsymbol{g}_t \odot \boldsymbol{g}_t$) exhibit high variance due to stochastic mini-batch sampling. The EMA mechanism acts as a low-pass filter, suppressing high-frequency noise to reveal the stable underlying importance trend. This formulation effectively approximates the local curvature:

$$S_t^{(j)} \approx \mathbb{E}_{t' \le t}\left[\left(\frac{\partial \mathcal{L}}{\partial \theta_j}\right)^2\right] \propto \mathbf{H}_{jj}. \tag{6}$$

2.3.2 Adaptive Layer-wise Normalization

Crucially, simply ranking $\boldsymbol{S}_t$ globally is flawed. In deep Transformer architectures, gradient magnitudes vary significantly across layers due to backpropagation dynamics (i.e., the gradient scale mismatch). Consequently, a global thresholding strategy would disproportionately bias the mask towards specific layers with naturally larger gradients, leaving other critical layers unprotected (as shown in Figure 4).

To ensure unbiased parameter selection, we apply Adaptive Min-Max Normalization. Let Ω_l denote the set of parameters in the l-th layer. We first compute layer-wise statistics:

$$\mu_{t,l} = \min_{k \in \Omega_l} S_t^{(k)}, \quad \nu_{t,l} = \max_{k \in \Omega_l} S_t^{(k)}. \tag{7}$$

The normalized score $\hat{I}_t^{(j)}$ for parameter $j \in \Omega_l$ is then computed as:

$$\hat{I}_t^{(j)} = \frac{S_t^{(j)} - \mu_{t,l}}{\nu_{t,l} - \mu_{t,l} + \epsilon}. \tag{8}$$

This projects importance scores into a unified $[0, 1]$ probability space, preserving the relative importance ranking within each layer while normalizing scale differences.

2.4 Dynamic Mask Generation and Evolution

Based on $\hat{\boldsymbol{I}}_t$, EPI periodically updates the isolation mask. We define an update interval H. At step t (where $t \bmod H = 0$), the binary mask $\boldsymbol{m}_t$ is generated via Top-$p\%$ selection:

$$m_t^{(j)} = \mathbb{I}\left(\hat{I}_t^{(j)} \ge \rho_t(p)\right), \tag{9}$$

where $\rho_t(p)$ is the dynamic threshold corresponding to the $(1-p)$-quantile.

Crucially, this mechanism enables the Evolution of Protection. We formally define the parameter state transitions as:

$$\mathcal{S}_{\text{lock}}^{(t)} = \{j \mid m_t^{(j)} = 1 \wedge m_{t-1}^{(j)} = 0\}, \tag{10}$$

$$\mathcal{S}_{\text{free}}^{(t)} = \{j \mid m_t^{(j)} = 0 \wedge m_{t-1}^{(j)} = 1\}. \tag{11}$$

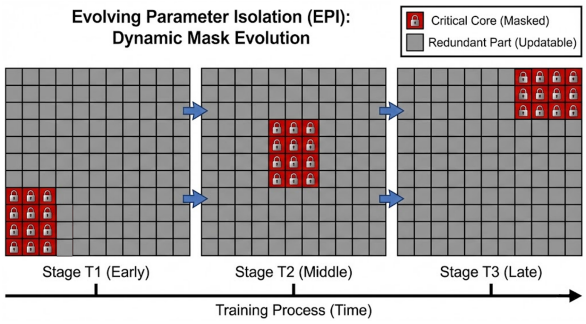

Figure 5: **Dynamic Mask Evolution.** Instead of a fixed constraint, the protection mask evolves periodically ($T_1 \rightarrow T_3$). Critical parameters (Red) are locked to prevent forgetting, while redundant ones (Grey) are released. This creates a moving shield that adapts to the temporal drift of parameter importance.

Here, $\mathcal{S}_{\text{lock}}$ represents emerging critical parameters (visualized as the Red Set in Figure 5), while $\mathcal{S}_{\text{free}}$ denotes parameters that became redundant and are released for plasticity.

2.5 Optimization with Dynamic Masking

To enforce isolation under AdamW, we apply the mask to the final parameter update (so protected parameters also receive no decoupled weight decay). At step t, we first compute the AdamW update:

$$\Delta\boldsymbol{\theta}_t \leftarrow \text{AdamWStep}(\boldsymbol{\theta}_t, \boldsymbol{g}_t), \tag{12}$$

and then update parameters with the isolation mask:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + (\mathbf{1} - \boldsymbol{m}_t) \odot \Delta\boldsymbol{\theta}_t, \tag{13}$$

where $\mathbf{1}$ is the all-ones vector and $\boldsymbol{m}_t \in \{0,1\}^d$. This blocks updates on protected parameters and mitigates forgetting (Eq. 3) while preserving adaptation capacity.

3 Experimental Setup

We evaluate EPI in a sequential multi-stage supervised fine-tuning setting, where heterogeneous tasks are learned stage by stage rather than jointly mixed within each mini-batch. We conduct extensive experiments to assess the effectiveness of EPI under this challenging scenario. Our evaluation is designed to answer the following questions: (i) whether EPI consistently outperforms standard supervised fine-tuning (SFT) and static parameter isolation baselines under heterogeneous and conflicting task distributions; (ii) whether dynamically evolving isolation masks are more effective than fixed masks in mitigating task interference and catastrophic forgetting; (iii) how sensitive EPI is to key hyperparameters such as the isolation ratio p and update interval H; and (iv) whether EPI maintains strong generalization across both structured reasoning and open-ended instruction-following tasks.

3.1 Datasets

We evaluate EPI on a diverse collection of publicly available benchmarks covering mathematical reasoning, logical reasoning, code generation, and open-ended instruction following. This heterogeneous task mixture is intentionally designed to induce parameter contention and expose interference effects.

Mathematical Reasoning. We use GSM8K (Cobbe et al., 2021), a widely adopted benchmark for multi-step arithmetic reasoning. Performance is measured using exact-match accuracy.

Logical Reasoning. We evaluate logical consistency using LogiQA (Liu et al., 2020), which requires multi-hop deductive reasoning over short passages. Accuracy is used as the evaluation metric.

Code Generation. For code synthesis, we adopt CodeAlpaca (Chaudhary, 2023), following prior work (Wang et al., 2025). We evaluate generation quality using CodeBLEU (Ren et al., 2020).

Instruction Following and Dialogue. We use Alpaca (Taori et al., 2023) and UltraChat (Ding et al., 2023) to assess general instruction-following and conversational abilities. Following established practice (Zheng et al., 2023), responses are evaluated using GPT-4-based scoring on a 1–10 scale.

Each dataset is evaluated using its standard metric. To facilitate a unified comparison across heterogeneous tasks, we additionally report a macro-average score (Avg. Norm. Score) by normalizing all task-specific metrics to a common 0–10 scale.

3.2 Baselines

We compare EPI against the following strong SFT baselines, largely following the experimental protocol of Wang et al. (2025) to ensure fairness and comparability.

Full Multi-task SFT. The model is fine-tuned on a uniform mixture of all datasets, updating all parameters jointly without task grouping or isolation. This represents the standard instruction tuning paradigm and serves as a lower bound to quantify the severity of task interference in the shared parameter space.

Base Model	Method	GSM8K	CodeAlpaca	LogiQA	Alpaca	UltraChat	Avg. Norm.
LLaMA-3-8B	Full SFT	55.6	28.3	60.8	7.8	8.1	7.32
	Multi-Stage (Random)	56.9	27.9	61.5	8.0	8.2	7.44
	Multi-Stage (Heuristic)	57.4	28.7	62.1	7.7	8.0	7.48
	Static Isolation ($p = 1\%$)	59.2	29.4	63.0	8.1	8.3	7.68
	EPI (Ours)	**61.8**	**31.2**	**65.4**	**8.4**	**8.6**	**7.98**
Mistral-7B	Full SFT	53.1	26.7	58.9	7.5	7.9	7.05
	Multi-Stage (Random)	54.4	26.3	59.6	7.7	8.0	7.17
	Multi-Stage (Heuristic)	55.0	27.1	60.2	7.4	7.8	7.21
	Static Isolation ($p = 1\%$)	56.8	27.9	61.4	7.8	8.1	7.44
	EPI (Ours)	**59.3**	**29.5**	**63.1**	**8.2**	**8.4**	**7.74**
Qwen2-7B	Full SFT	57.2	29.6	62.4	8.0	8.3	7.55
	Multi-Stage (Random)	58.5	29.2	63.0	8.2	8.4	7.67
	Multi-Stage (Heuristic)	59.1	30.0	63.7	7.9	8.2	7.71
	Static Isolation ($p = 1\%$)	61.0	30.8	64.9	8.3	8.6	7.93
	EPI (Ours)	**63.7**	**32.6**	**67.2**	**8.7**	**8.9**	**8.23**
Gemma-2-9B	Full SFT	58.5	30.1	64.0	8.3	8.5	7.78
	Multi-Stage (Random)	59.8	29.8	64.6	8.5	8.7	7.90
	Multi-Stage (Heuristic)	60.4	30.6	65.2	8.2	8.4	7.97
	Static Isolation ($p = 1\%$)	62.2	31.5	66.4	8.6	8.8	8.21
	EPI (Ours)	**65.0**	**33.2**	**69.1**	**8.9**	**9.1**	**8.57**

Table 1: **Main experimental results on heterogeneous SFT benchmarks.** EPI consistently outperforms standard SFT, multi-stage training, and static parameter isolation across all base models. All values are averaged over three independent runs; detailed mean $\pm$ standard deviation results are provided in Appendix E.1. Background colors denote the top-3 methods (darker orange indicates higher performance).

Multi-Stage SFT (Random Grouping). Tasks are randomly partitioned into separate stages. The model is fine-tuned sequentially on each stage, updating all parameters throughout training. This setting simulates a basic sequential learning scenario, allowing us to evaluate the model's vulnerability to catastrophic forgetting when task order is randomized.

Multi-Stage SFT (Heuristic Grouping). Tasks are grouped based on high-level semantic similarity (e.g., reasoning tasks vs. open-ended dialogue) and fine-tuned sequentially, with all parameters updated in each stage. This baseline tests whether manual curriculum design can mitigate interference more effectively than random ordering, or if parameter isolation is necessary.

Static Parameter Isolation. We implement static parameter isolation following Sorrenti et al. (2023); Wang et al. (2025), where a fixed subset of task-critical parameters is identified during an initial probing phase and frozen throughout subsequent training. Crucially, this method relies on a pre-computed mask derived from initial statistics, assuming that parameter importance remains invariant to the optimization trajectory—an assumption directly challenged by our EPI framework.

3.3 Implementation Details

We evaluate all methods using recent, strong open-weight LLMs as backbone models, updating earlier baselines to reflect current practice. Specifically, we use LLaMA-3-8B (Grattafiori et al., 2024), Qwen2-7B (Yang et al., 2024), Gemma-2-9B (Team et al., 2024) and Mistral-7B (Jiang et al., 2023) as base models. All models are fine-tuned using the AdamW optimizer (Loshchilov and Hutter, 2017) with a learning rate of 1×10^{-5}, batch size of 64, and a cosine learning rate schedule with 3% warm-up steps. Unless otherwise stated, all SFT procedures are conducted for three epochs. For static isolation baselines, task-critical parameters are identified via probe fine-tuning runs of one epoch per task, following Wang et al. (2025). The isolation ratio is set to $p = 1\%$. For EPI, importance statistics are updated online using exponential moving averages of squared gradients (Eq. 5). Isolation masks are refreshed every $H = 500$ optimization steps, with a default isolation ratio of

$p = 1\%$. All experiments are conducted on eight NVIDIA A100 GPUs (80GB). To ensure reproducibility and robustness, we conduct each experiment over three independent runs with different random seeds and report the average performance.

3.4 Main Results

Table 1 summarizes the performance of EPI compared to standard SFT and various isolation baselines across four state-of-the-art backbone models. The results support three key observations:

EPI consistently establishes a new state-of-the-art across heterogeneous benchmarks. As shown in Table 1, EPI achieves the highest Average Normalized Score across all four base models, consistently outperforming the standard Full SFT baseline by a significant margin (e.g., +0.79 average score improvement on Gemma-2-9B). Notably, simple Multi-Stage SFT strategies (both Random and Heuristic) often fail to yield consistent improvements over Full SFT, and in some cases (e.g., Mistral-7B on CodeAlpaca), sequential training even degrades performance due to catastrophic forgetting. This confirms that merely reordering data or strictly separating training stages is insufficient to resolve parameter contention in the shared parameter space.

Dynamic evolution outperforms static protection, especially on complex reasoning. The most critical comparison lies between EPI and Static Isolation, as both methods aim to protect task-critical parameters. Our results demonstrate that EPI consistently surpasses Static Isolation across all tasks. Crucially, the performance gap is most pronounced in reasoning-intensive tasks. For instance, on GSM8K, EPI outperforms Static Isolation by +2.6% on LLaMA-3-8B and +2.8% on Gemma-2-9B. We attribute this to the fact that reasoning capabilities often require the formation of complex functional circuits that evolve during training. Static masks, fixed at initialization, fail to capture these emerging dependencies. By contrast, EPI's ability to dynamically lock emerging critical parameters allows the model to secure new reasoning pathways as they are learned.

EPI effectively mitigates the Stability-Plasticity dilemma. A common failure mode in static parameter isolation is over-protection, where locking too many parameters hinders the learning of new open-ended tasks (Plasticity). However, EPI achieves superior performance not only on rigid reasoning tasks but also on creative generation tasks (Alpaca, UltraChat). For example, on Qwen2-7B, EPI improves UltraChat performance to 8.9, surpassing both Full SFT (8.3) and Static Isolation (8.6). This indicates that EPI successfully releases outdated parameters (as detailed in Sec. 2), recovering sufficient capacity to adapt to open-ended instructions without overwriting established knowledge. This balance is reflected in the Average Normalized Score, where EPI demonstrates the most robust generalization capability across diverse task categories.

4 Ablation Study

To disentangle the contributions of individual components within the EPI framework, we conduct a comprehensive ablation study using LLaMA-3-8B and Gemma-2-9B. We systematically investigate the following key aspects: (1) the necessity of dynamic evolution and layer-wise normalization; (2) the superiority of our selection mechanism over alternative strategies; (3) the impact of temporal gradient smoothing via EMA; (4) the sensitivity to critical hyperparameters, specifically the isolation ratio p and the mask-update interval H; and (5) the underlying temporal and structural dynamics of the evolving masks. Unless otherwise stated, we report the Avg. Norm. Score to facilitate a unified comparison across heterogeneous tasks.

4.1 Component Effectiveness Analysis

We first validate the core premise of EPI: namely, that parameter isolation must be dynamic and normalized. Table 2 compares the full EPI method against a static baseline and a variant without layer-wise normalization.

Mask Strategy	LLaMA-3	Gemma-2
Static Mask ($p = 1\%$)	7.68	8.21
Static Mask + Layer Norm	7.82	8.36
EPI (w/o Layer Norm)	7.83	8.35
EPI (Full)	**7.98**	**8.57**

Table 2: **Ablation of mask strategy.** Both layer-wise normalization and dynamic evolution contribute meaningfully, and their combination yields the strongest performance.

Complementary roles of dynamic evolution and layer-wise normalization. Table 2 shows that both components contribute meaningfully and are complementary. Adding layer-wise normalization

to a static mask already yields clear gains on both backbones (LLaMA-3: 7.68 $\rightarrow$ 7.82; Gemma-2: 8.21 $\rightarrow$ 8.36), confirming that normalization alleviates cross-layer gradient-scale bias even without mask evolution. Conversely, enabling dynamic evolution without layer normalization also improves performance over the static baseline (LLaMA-3: 7.68 $\rightarrow$ 7.83; Gemma-2: 8.21 $\rightarrow$ 8.35), showing that periodic re-selection is itself effective in tracking temporal drift.

Why the full design works best. The strongest results are achieved only when both components are used together, reaching 7.98 on LLaMA-3 and 8.57 on Gemma-2. This indicates that the gains from EPI do not come from a single design choice alone. Instead, layer-wise normalization improves the quality and fairness of importance scores across layers, while dynamic evolution improves their temporal adaptivity throughout training. Their combination yields the most effective protection mechanism under heterogeneous SFT.

4.2 Alternative Selection Strategies

To better justify our normalization and selection design, we compare EPI against two simpler alternatives. The first is Per-Layer Fixed Budget, which selects the top-$p\%$ parameters independently within each layer. The second is Global Raw Scores, which ranks all parameters globally using unnormalized importance scores. Table 3 summarizes the results on LLaMA-3-8B.

Strategy	Avg. Norm. Score
Per-Layer Fixed Budget	7.82
Global Raw Scores	7.71
EPI (Ours)	**7.98**

Table 3: **Comparison of alternative allocation strategies on LLaMA-3-8B.** Both per-layer fixed quotas and global raw ranking underperform our layer-wise normalization plus global top-$p\%$ selection.

Both simpler alternatives are inferior to the full EPI design. Per-layer fixed budgets impose a rigid quota regardless of each layer's actual sensitivity and plasticity, which can lead to misallocation of the protection budget. By contrast, global raw ranking is biased toward layers with naturally larger gradient magnitudes and therefore over-protects high-scale layers. EPI performs best because it first removes cross-layer scale bias through layer-wise normalization and then allocates the protection budget globally, allowing protection to concentrate where it is most needed.

4.3 Impact of Gradient Smoothing (EMA)

Smoothing Factor	LLaMA-3	Gemma-2
No EMA (Instantaneous)	7.81	8.39
EMA ($\beta = 0.9$)	7.90	8.50
EMA ($\beta = 0.99$)	**7.98**	**8.57**

Table 4: **Effect of Temporal Accumulation.** A high smoothing factor ($\beta = 0.99$) is essential to filter stochastic noise and reveal stable importance trends.

The results indicate that raw instantaneous gradients ($\beta = 0$) are insufficient for reliable parameter isolation due to high variance. Notably, performance peaks at $\beta = 0.99$, which empirically validates that strong temporal smoothing is essential to filter out transient fluctuations and reveal consistent importance trends. This confirms that the isolation mechanism benefits from capturing the long-term optimization trajectory rather than reacting to stochastic gradient noise.

4.4 Sensitivity to Isolation Ratio (p)

Finally, we investigate the crucial trade-off between Stability and Plasticity by varying the core parameter isolation ratio p. The results are visualized in Figure 6.

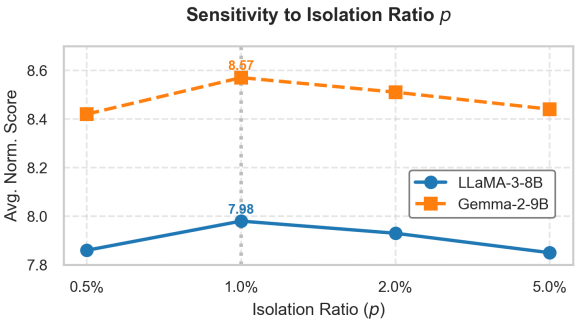

Figure 6: **Sensitivity Analysis to Isolation Ratio (p).** The plot illustrates the average normalized score across different isolation ratios for LLaMA-3 and Gemma-2. Both models exhibit a distinct peak performance at $p = 1.0\%$, representing an optimal trade-off between retaining prior knowledge (Stability) and adapting to new tasks (Plasticity).

As illustrated in Figure 6, we observe a clear inverted U-shaped trend consistent across both model

architectures. Performance peaks significantly at the critical value of $p = 1.0\%$, where LLaMA-3 achieves a score of 7.98 and Gemma-2 reaches 8.57.

Deviating from this optimal operating point leads to diminished returns, highlighting the delicate balance required. Lower ratios ($p = 0.5\%$) result in a shield too sparse to ensure stability against forgetting, whereas higher ratios ($p \geq 2.0\%$) introduce excessive rigidity that limits the plasticity needed for new task adaptation. This finding, where the optimal balance is struck at a mere $p = 1\%$, provides evidence that task-critical knowledge in LLMs is highly sparse, residing in only a very small fraction of important parameters.

4.5 Sensitivity to Mask-Update Interval

Although the importance statistic S_t is updated continuously, the binary protection mask is refreshed only every H steps. This design trades off responsiveness against computational overhead and optimization stability. Very frequent updates can introduce threshold thrashing for parameters near the top-$p\%$ boundary, while overly infrequent updates lag behind temporal drift.

Update Interval (H)	Avg. Norm. Score
100	7.92
500	**7.98**
1000	7.95
2000	7.89

Table 5: **Sensitivity to the mask-update interval H on LLaMA-3-8B.** $H = 500$ provides the best trade-off between responsiveness and stability.

As shown in Table 5, performance is robust across a reasonable range of update intervals, with a slight peak at $H = 500$. This supports our design choice of periodic rather than per-step mask updates: updating too frequently introduces instability, while updating too slowly reduces adaptability to drifting parameter importance.

4.6 Quantifying Mask Dynamics

To verify the existence of Parameter Importance Drift, we track the evolution of isolation masks throughout the training process.

Temporal Divergence. We first examine the stability of parameter importance over time. Concretely, on LLaMA-3-8B, the Jaccard overlap between the initial mask and the final mask drops from 78.3% at 25% training progress to 44.7% at the end of training, indicating that more than half of the initially protected parameters are replaced over time (see Appendix E.2). This divergence indicates that many parameters deemed critical at initialization become redundant as the model adapts. Consequently, static isolation approaches, which enforce a rigid lock based on initial statistics, inevitably waste capacity on outdated parameters while failing to secure newly emerging functional circuits.

Structural and Task Adaptation. Beyond global drift, the mask evolution exhibits clear structural patterns. We observe that upper layers (closer to the output) show significantly faster evolution rates compared to lower layers (see Appendix E.2). This aligns with the intuition that upper layers manage complex, task-specific reasoning which requires high plasticity, while lower layers handle stable linguistic features. Furthermore, the mask shows distinct reconfiguration patterns immediately following task transitions, confirming that EPI is actively responsive to data distribution shifts rather than drifting randomly.

5 Conclusion

In this paper, we investigated the dynamic nature of parameter importance during supervised fine-tuning (SFT) of large language models and identified the limitations of static isolation methods. We proposed **Evolving Parameter Isolation (EPI)**, a novel framework that continuously tracks parameter sensitivity via gradient-based online estimation, applies adaptive layer-wise normalization, and evolves protection masks to dynamically align with the learning trajectory. Extensive experiments across multiple LLM backbones and heterogeneous tasks including reasoning, code generation, and instruction following demonstrated that EPI consistently outperforms standard SFT and static isolation baselines. Our analyses show that EPI not only improves overall task performance but also significantly reduces catastrophic forgetting and gradient interference (the seesaw effect), highlighting the importance of synchronizing isolation mechanisms with the evolving optimization dynamics of SFT. Our findings suggest that future fine-tuning paradigms must move beyond rigid static constraints and embrace dynamic, adaptive mechanisms to fully unlock the potential of LLMs in heterogeneous multi-task scenarios.

6 Limitations

Despite its strong empirical performance, EPI has several limitations that should be considered in future work.

First, while the online estimation of parameter importance using EMA of squared gradients is computationally efficient, it introduces additional memory overhead proportional to the model size. For extremely large models (e.g., 30B+ parameters), this could limit scalability without further optimization or sparsity-aware techniques.

Second, EPI currently relies on heuristic hyperparameters such as the core percentage p and mask update interval H. Although we show robustness across a range of values, these parameters may require careful tuning for different models or task distributions. An adaptive mechanism to auto-tune p based on loss landscapes would be a valuable extension.

Third, our experiments focus primarily on English-centric datasets and structured reasoning, code, and dialogue tasks. The effectiveness of EPI on low-resource languages, highly multimodal data, or continual adaptation with streaming data remains to be validated.

Finally, while dynamic isolation mitigates gradient interference, it does not explicitly model task similarity or hierarchy. Future extensions could incorporate task-aware importance propagation or knowledge distillation techniques to further enhance cross-task generalization.

References

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Prakash Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2021. Ext5: Towards extreme multi-task scaling for transfer learning. *CoRR*, abs/2111.10952.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2025. A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*.

Sahil Chaudhary. 2023. Code alpaca: An instruction-following llama model for code generation.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pages 794–803. PMLR.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, and 16 others. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Chang Dai, Hongyu Shan, Mingyang Song, and Di Liang. 2025. Hope: Hyperbolic rotary positional encoding for stable long-range dependency modeling in large language models. *arXiv preprint arXiv:2509.05218*.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2024. How abilities in large language models are affected by supervised fine-tuning data composition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 177–198.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. 2020. Rigging the lottery: Making all tickets winners. In *International conference on machine learning*, pages 2943–2952. PMLR.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Zichu Fei, Qi Zhang, Tao Gui, Di Liang, Sirui Wang, Wei Wu, and Xuan-Jing Huang. 2022. Cqg: A simple and effective controlled generation framework for multi-hop question generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6896–6906.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

Ziyuan Gao, Di Liang, Xianjie Wu, Philippe Morel, and Minlong Peng. 2026. Decorl: Decoupling reasoning chains via parallel sub-step generation and cascaded reinforcement for interpretable and scalable rlhf. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 30789–30797.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

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 *International conference on machine learning*, pages 2790–2799. PMLR.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.

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 *International Conference on Learning Representations*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Bo Li, Di Liang, and Zixin Zhang. 2024a. Comateformer: Combined attention transformer for semantic sentence matching. *arXiv preprint arXiv:2412.07220*.

Liang Li, Qisheng Liao, Meiting Lai, Di Liang, and Shangsong Liang. 2024b. Local and global: Text matching via syntax graph calibration. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11571–11575. IEEE.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947.

Di Liang, Fubao Zhang, Qi Zhang, and Xuan-Jing Huang. 2019. Asynchronous deep interaction network for natural language inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2692–2700.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

Xiaoyu Liu, Xiaoyu Guan, Di Liang, and Xianjie Wu. 2026. Dpi: Exploiting parameter heterogeneity for interference-free fine-tuning. *arXiv preprint arXiv:2601.17777*.

Xiaoyu Liu, Di Liang, Hongyu Shan, Peiyang Liu, Yonghao Liu, Muling Wu, Yuntao Li, Xianjie Wu, Li Miao, Jiangrong Shen, and Minlong Peng. 2025. Structural reward model: Enhancing interpretability, efficiency, and scalability in reward modeling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 672–685, Suzhou (China). Association for Computational Linguistics.

Yonghao Liu, Mengyu Li, Di Liang, Ximing Li, Fausto Giunchiglia, Lan Huang, Xiaoyue Feng, and Renchu Guan. 2024. Resolving word vagueness with scenario-guided adapter for natural language inference. *arXiv preprint arXiv:2405.12434*.

Yonghao Liu, Di Liang, Fang Fang, Sirui Wang, Wei Wu, and Rui Jiang. 2023a. Time-aware multiway adaptive fusion network for temporal knowledge graph question answering. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Yonghao Liu, Di Liang, Mengyu Li, Fausto Giunchiglia, Ximing Li, Sirui Wang, Wei Wu, Lan Huang, Xiaoyue Feng, and Renchu Guan. 2023b. Local and global: Temporal question answering via information fusion. In *IJCAI*, pages 5141–5149.

David López-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In

Advances in Neural Information Processing Systems (NeurIPS).

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Jinghui Lu, Dongsheng Zhu, Weidong Han, Rui Zhao, Brian Mac Namee, and Fei Tan. 2023. What makes pre-trained language models better zero-shot learners? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2288–2303.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. 2023. Wizardcoder: Empowering code large language models with evol-instruct. *arXiv preprint arXiv:2306.08568*.

Arun Mallya and Svetlana Lazebnik. 2018. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773.

James Martens. 2020. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Siyuan Mu and Sen Lin. 2025. A comprehensive survey of mixture-of-experts: Algorithms, theory, and applications. *arXiv preprint arXiv:2503.07137*.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. 2014. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Jupinder Parmar, Sanjev Satheesh, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Reuse, don't retrain: A recipe for continued pretraining of language models. *arXiv preprint arXiv:2407.07263*.

Razvan Pascanu and Yoshua Bengio. 2013. Revisiting natural gradient for deep networks. *arXiv preprint arXiv:1301.3584*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th conference of the European chapter of the association for computational linguistics: main volume*, pages 487–503.

Zhen Qi, Jiajing Chen, Shuo Wang, Bingying Liu, Hongye Zheng, and Chihang Wang. 2024. Optimizing multi-task learning for enhanced performance in large language models. In *2024 4th International Conference on Electronic Information Engineering and Computer Communication (EIECC)*, pages 1179–1183. IEEE.

Shuo Ren, Daya Guo, Shuai Lu, Long Zhou, Shujie Liu, Duyu Tang, Neel Sundaresan, Ming Zhou, Ambrosio Blanco, and Shuai Ma. 2020. Codebleu: a method for automatic evaluation of code synthesis. *arXiv preprint arXiv:2009.10297*.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. 2021. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. *Advances in neural information processing systems*, 32.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M. Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, and 22 others. 2021. Multitask prompted training enables zero-shot task generalization. *CoRR*, abs/2110.08207.

Jian Song, Di Liang, Rumei Li, Yuntao Li, Sirui Wang, Minlong Peng, Wei Wu, and Yongxin Yu. 2022. Improving semantic matching through dependency-enhanced pre-trained model with adaptive fusion. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 45–57.

Amelia Sorrenti, Giovanni Bellitto, Federica Proietto Salanitri, Matteo Pennisi, Concetto Spampinato, and Simone Palazzo. 2023. Selective freezing for efficient continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3550–3559.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995. PMLR.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, and 179 others. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Rongzheng Wang, Yihong Huang, Muquan Li, Jiakai Li, Di Liang, Bob Simons, Pei Ke, Shuang Liang, and Ke Qin. 2026. Rethinking llm-driven heuristic design: Generating efficient and specialized solvers via dynamics-aware optimization. *arXiv preprint arXiv:2601.20868*.

Sirui Wang, Di Liang, Jian Song, Yuntao Li, and Wei Wu. 2022. Dabert: Dual attention enhanced bert for semantic matching. In *Proceedings of the 29th international conference on computational linguistics*, pages 1645–1654.

Yao Wang, Di Liang, and Minlong Peng. 2025. Not all parameters are created equal: Smart isolation boosts fine-tuning performance. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9882–9896.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 13484–13508.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Muling Wu, Qi Qian, Wenhao Liu, Xiaohua Wang, Zisu Huang, Di Liang, LI Miao, Shihan Dou, Changze Lv, Zhenghua Wang, Zhibo Xu, Lina Chen, Tianlong Li, Xiaoqing Zheng, and Xuanjing Huang. 2025a. Progressive mastery: Customized curriculum learning with guided prompting for mathematical reasoning. *Preprint*, arXiv:2506.04065.

Xianjie Wu, Di Liang, Jian Yang, Xianfu Cheng, LinZheng Chai, Tongliang Li, Liqun Yang, and Zhoujun Li. 2025b. Breaking size barrier: Enhancing reasoning for large-size table question answering. In *International Conference on Database Systems for Advanced Applications*, pages 241–256. Springer.

Xianjie Wu, Xiaohang Xu, Tingyu Jiang, Jian Yang, Di Liang, Xianfu Cheng, Zhenhe Wu, Linzheng Chai, Wei Zhang, Jiaheng Liu, Ge Zhang, Bob Simons, Tongliang Li, and Zhoujun Li. 2026. Mmtablebench: A multi-level multimodal benchmark for reasoning and layout complexity in table qa. In *Proceedings of the ACM Web Conference 2026*, WWW '26, page 3881–3892, New York, NY, USA. Association for Computing Machinery.

Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, Tongliang Li, Zhoujun Li, and Guanglin Niu. 2025c. Tablebench: a comprehensive and complex benchmark for table question answering. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press.

Xianjie Wu, Jian Yang, Tongliang Li, Shiwei Zhang, Yiyang Du, LinZheng Chai, Di Liang, and Zhoujun Li. 2025d. Unleashing potential of evidence in knowledge-intensive dialogue generation. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Chao Xue, Di Liang, Pengfei Wang, and Jing Zhang. 2024. Question calibration and multi-hop modeling for temporal question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19332–19340.

Chao Xue, Di Liang, Sirui Wang, Jing Zhang, and Wei Wu. 2023. Dual path modeling for semantic matching by perceiving subtle conflicts. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in neural information processing systems*, 33:5824–5836.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In

Proceedings of the 37th International Conference on Neural Information Processing Systems, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Rui Zheng, Bao Rong, Yuhao Zhou, Di Liang, Sirui Wang, Wei Wu, Tao Gui, Qi Zhang, and Xuan-Jing Huang. 2022. Robust lottery tickets for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2211–2224.

A Related Work

A.1 Challenges in Heterogeneous SFT

Supervised Fine-Tuning (SFT) has become a core paradigm for adapting pre-trained large language models (LLMs) to downstream instructions and tasks (Brown et al., 2020; Minaee et al., 2024; Vaswani et al., 2017; Howard and Ruder, 2018). Building on this foundation, recent instruction-tuning approaches have substantially improved the generalization ability of LLMs across diverse tasks (Wei et al., 2021; Sanh et al., 2021; Chung et al., 2022; Wang et al., 2023; Gao et al., 2026; Ouyang et al., 2022). Meanwhile, the rapid expansion of model scale, data scale, and task diversity has further broadened the scope of capabilities that LLMs are expected to support (Lu et al., 2023; Luo et al., 2023; Liu et al., 2024, 2025).

A major challenge, however, is that modern SFT must accommodate an increasingly heterogeneous capability space within a single shared parameter space. Compared with earlier NLP settings centered on semantic matching and natural language inference (Wang et al., 2022; Xue et al., 2023; Song et al., 2022; Li et al., 2024b; Liang et al., 2019; Li et al., 2024a), contemporary LLMs must master a multifaceted continuum of intelligence, encompassing strict formal reasoning (Wu et al., 2025b,c, 2026; Wang et al., 2026), complex long-chain deduction (Xue et al., 2024; Liu et al., 2023a,b; Fei et al., 2022), and versatile open-domain generation (Dai et al., 2025; Wu et al., 2025d). These capabilities differ markedly in input structure, output format, and optimization objective, making them difficult to reconcile through uniform parameter updates.

As a result, SFT over heterogeneous task streams often suffers from gradient conflict and update competition. Updates that improve one capability may degrade another, leading to the "seesaw phenomenon" (Yu et al., 2020), shifts in ability composition under different fine-tuning data mixtures (Dong et al., 2024), and eventually catastrophic forgetting (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017).

A.2 Traditional Mitigation Paradigms

To address these conflicts, prior literature has predominantly explored two directions: global optimization within shared spaces and physical separation via modular architectures.

Global Optimization in Shared Parameter Spaces. Strategies in this category attempt to reconcile conflicts by globally adjusting the optimization trajectory of the entire model, treating parameters as a unified and fully shared entity. Data-centric approaches utilize heuristic curriculum learning (Wei et al., 2021; Wu et al., 2025a) or dynamic scheduling (Aribandi et al., 2021) to balance task exposure. To mitigate forgetting in sequential settings, replay mechanisms like DMT (Dong et al., 2024) and experience replay (Rolnick et al., 2019) introduce historical data into current training stages. Complementarily, gradient-based methods constrain update directions. Regularization techniques such as Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017; Li and Hoiem, 2017) penalize changes to important weights, while projection-based gradient constraints (or gradient surgery) such as GEM (López-Paz and Ranzato, 2017) and PCGrad (Yu et al., 2020) modify gradients to reduce conflicts. Separately, gradient normalization methods such as GradNorm (Chen et al., 2018) adaptively reweight tasks by matching gradient magnitudes. However, finding a single update direction for conflicting objectives remains geometrically infeasible, failing to resolve destructive interference.

Architectural Modularization. To avoid contention in shared spaces, a distinct paradigm physically decouples task representations by expanding the model architecture. This includes Mixture-of-Experts (MoE) (Fedus et al., 2022; Riquelme et al., 2021), Adapter Fusion (Pfeiffer et al., 2021), and split-architecture designs (Stickland and Murray, 2019). While effective at reducing interference through separation, these modular approaches introduce significant structural complexity and memory overhead. Furthermore, manually assigning static modules often fails to scale efficiently to hundreds of fine-grained capabilities, potentially leading to knowledge fragmentation (Cai et al., 2025).

Base Model	Method	Forgetting Ratio (FR, %) ↓					Avg. FR
		GSM8K	Code	LogiQA	Alpaca	UChat	
LLaMA-3-8B	Full SFT (Multi-task)	12.6	10.4	9.8	7.5	6.9	9.44
	Multi-Stage (Random)	9.7	8.9	8.1	6.8	6.2	7.94
	Multi-Stage (Heuristic)	8.9	8.2	7.5	6.5	6.0	7.42
	Static Isolation ($p = 1\%$)	6.1	5.8	5.2	4.5	4.1	5.14
	EPI (Ours, $p = 1\%$)	**5.3**	**5.1**	**4.8**	**4.2**	**3.9**	**4.66**
Gemma-2-9B	Full SFT (Multi-task)	11.2	9.7	9.0	7.3	7.1	8.66
	Multi-Stage (Random)	8.5	7.9	7.3	6.5	6.0	7.04
	Multi-Stage (Heuristic)	7.9	7.2	6.8	6.2	5.7	6.36
	Static Isolation ($p = 1\%$)	5.6	5.3	4.9	4.1	3.8	4.74
	EPI (Ours, $p = 1\%$)	**4.8**	**4.5**	**4.2**	**3.8**	**3.4**	**4.14**
Qwen2-7B	Full SFT (Multi-task)	13.0	11.5	10.2	8.0	7.5	10.04
	Multi-Stage (Random)	10.1	9.1	8.5	6.9	6.4	8.20
	Multi-Stage (Heuristic)	9.4	8.5	8.0	6.7	6.1	7.74
	Static Isolation ($p = 1\%$)	6.8	6.3	5.8	4.9	4.5	5.66
	EPI (Ours, $p = 1\%$)	**5.7**	**5.0**	**4.7**	**4.1**	**3.8**	**4.66**

Table 6: Forgetting analysis across tasks and base models. Lower FR (%) indicates better retention of previously learned knowledge. EPI consistently reduces forgetting compared to standard SFT and static isolation baselines.

A.3 Parameter-Level Isolation Techniques

Recent research has pivoted towards a third paradigm: Parameter-Level Isolation. Grounded in the theory of parameter heterogeneity (Neyshabur et al., 2014; Frankle and Carbin, 2018; Mu and Lin, 2025), these methods aim to construct logical subspaces within the native backbone, offering a balance between the efficiency of shared models and the stability of modular designs.

Implicit Subspaces via Additive Modules. Parameter-Efficient Fine-Tuning (PEFT) methods, exemplified by Adapters (Houlsby et al., 2019; Li and Liang, 2021) and LoRA (Hu et al., 2022), implicitly achieve isolation by freezing the backbone and optimizing task-specific low-rank matrices. While this effectively separates task knowledge, it fundamentally relies on additive parameters. This design choice leaves the vast majority of the pre-trained dense knowledge dormant, potentially limiting the model's capacity to leverage its full reasoning power during adaptation.

Explicit Partitioning and Static Isolation. Distinct from additive methods, explicit isolation strategies partition the existing parameters of the pre-trained backbone. Early works in transfer learning explored iterative pruning to specialize weights (Mallya and Lazebnik, 2018; Parmar et al., 2024; Zheng et al., 2022). More recently, Core Parameter Isolation Fine-Tuning (CPI-FT) (Wang et al., 2025) advanced this concept by explicitly identifying and freezing task-critical parameters based on their update magnitudes. However, a critical limitation persists in these approaches: they rely on a static assumption. Current methods typically fix the isolation mask at initialization, ignoring the temporal dynamics of the optimization process. As training progresses and feature hierarchies reorganize, parameters initially deemed critical may become redundant, while previously unimportant parameters may emerge as essential (Liu et al., 2026; Evci et al., 2020).

Evolving Parameter Isolation (Ours). Our proposed EPI framework addresses this gap by transitioning from static to dynamic isolation. Unlike coarse-grained routing (Qi et al., 2024) or static freezing mechanisms (Wang et al., 2025), EPI utilizes online gradient signals to continuously update the protection mask. By synchronizing the isolation mechanism with the model's evolving learning trajectory, EPI ensures that emerging capabilities are protected in real-time while maintaining the flexibility of the shared parameter space.

B Forgetting and Seesaw Effect Analysis

Catastrophic forgetting and task interference constitute fundamental challenges in multi-task supervised fine-tuning (SFT) of large language models (LLMs). When models undergo sequential or simultaneous fine-tuning on heterogeneous tasks, parameter updates intended for a specific task often degrade previously acquired capabilities—a phenomenon manifesting as the "seesaw effect". To mitigate these issues, our proposed Evolving Parameter Isolation (EPI) explicitly identifies and safeguards task-critical parameters dynamically, while permitting redundant parameters to adapt

Base Model	Method	Task Gradient Conflict (TGC) ↓					Avg. TGC
		Code	LogiQA	Alpaca	Code-Logi	Code-Alp	
LLaMA-3-8B	Full SFT (Multi-task)	0.42	0.38	0.36	0.33	0.30	0.358
	Multi-Stage (Random)	0.31	0.28	0.26	0.24	0.22	0.262
	Multi-Stage (Heuristic)	0.28	0.25	0.23	0.21	0.20	0.234
	Static Isolation ($p = 1\%$)	0.18	0.15	0.13	0.12	0.11	0.138
	EPI (Ours, $p = 1\%$)	**0.15**	**0.12**	**0.10**	**0.09**	**0.08**	**0.108**
Gemma-2-9B	Full SFT (Multi-task)	0.39	0.36	0.33	0.30	0.28	0.332
	Multi-Stage (Random)	0.28	0.26	0.23	0.22	0.20	0.238
	Multi-Stage (Heuristic)	0.25	0.23	0.21	0.19	0.18	0.212
	Static Isolation ($p = 1\%$)	0.14	0.12	0.11	0.09	0.08	0.108
	EPI (Ours, $p = 1\%$)	**0.13**	**0.11**	**0.09**	**0.08**	**0.07**	**0.096**
Qwen2-7B	Full SFT (Multi-task)	0.44	0.40	0.37	0.35	0.32	0.376
	Multi-Stage (Random)	0.33	0.30	0.27	0.25	0.22	0.274
	Multi-Stage (Heuristic)	0.30	0.27	0.25	0.23	0.20	0.250
	Static Isolation ($p = 1\%$)	0.17	0.15	0.13	0.11	0.10	0.132
	EPI (Ours, $p = 1\%$)	**0.16**	**0.13**	**0.11**	**0.09**	**0.08**	**0.114**

Table 7: Seesaw effect analysis via Task Gradient Conflict (TGC). Lower TGC signifies reduced task interference. EPI minimizes gradient conflicts more effectively than full multi-task and static baselines.

to novel tasks.

B.1 Evaluation Metrics

We employ two primary metrics to systematically quantify forgetting and task interference:

Forgetting Ratio (FR). For a given task $\mathcal{T}_i$, FR quantifies the degradation in performance on previously learned tasks following fine-tuning on new tasks:

$$\text{FR}_i = \frac{\text{Perf}_{\text{initial}}(\mathcal{T}_i) - \text{Perf}_{\text{after new tasks}}(\mathcal{T}_i)}{\text{Perf}_{\text{initial}}(\mathcal{T}_i)} \times 100\%. \tag{14}$$

Task Gradient Conflict (TGC). TGC measures the severity of gradient interference between any task pair $\mathcal{T}_i$ and $\mathcal{T}_j$:

$$\text{TGC}_{i,j} = \max\big(0, -\cos(\boldsymbol{g}_i, \boldsymbol{g}_j)\big), \tag{15}$$

where $\boldsymbol{g}_i$ and $\boldsymbol{g}_j$ denote the gradient vectors of the respective tasks. A higher TGC value indicates stronger interference, reflecting a pronounced seesaw effect.

We conduct sequential fine-tuning experiments starting with GSM8K (mathematical reasoning), followed by CodeAlpaca (code generation), LogiQA (logical reasoning), and finally Alpaca and UltraChat (instruction-following and dialogue). This sequence establishes a controlled environment to assess catastrophic forgetting on early tasks and gradient conflicts across diverse stages. We evaluate three backbones: LLaMA-3-8B, Gemma-2-9B, and Qwen2-7B. Baselines include Full Multi-task SFT, Multi-Stage Random grouping, and Multi-Stage Heuristic grouping. All models are trained under identical optimization and batch configurations to ensure fair comparison.

B.2 Results on Forgetting and Seesaw Effect

The experimental results, summarized in Table 6 and Table 7, yield several key observations. First, Full Multi-task SFT suffers from substantial forgetting and high gradient conflicts, attributable to indiscriminate parameter updates across heterogeneous tasks. Second, while Multi-stage baselines partially mitigate forgetting by isolating training stages, static grouping strategies fail to account for the emergence of new critical parameters, leaving sensitive regions vulnerable to interference. Third, EPI substantially lowers both FR and TGC across all backbones. This demonstrates that dynamically adapting isolation masks effectively preserves previously learned capabilities while facilitating the acquisition of new tasks.

Notably, Gemma-2-9B exhibits consistently lower FR and TGC compared to LLaMA-3-8B. This suggests that larger, more capable models benefit disproportionately from dynamic isolation, likely due to their higher capacity to represent multiple task-specific subspaces. The sequential evolution of the mask in EPI allows the model to track temporal drift in parameter importance, reducing the unnecessary locking of outdated parameters while ensuring emerging critical regions are protected. This dynamic balancing of stability and plasticity directly addresses the seesaw effect, en-

suring robust performance in multi-task settings.

C Case Study

To provide a qualitative understanding of how our method improves SFT, Table 8 presents representative cases across reasoning, coding, and dialogue tasks. While Full Multi-task SFT often suffers from reasoning inconsistencies due to task interference, and Static Mask Isolation occasionally yields suboptimal efficiency owing to its rigid parameter locking, EPI consistently maintains prior knowledge while adapting effectively to new demands. These examples intuitively demonstrate that dynamic parameter isolation successfully balances stability and plasticity, translating to superior contextual accuracy and structural efficiency.

D Algorithm and Complexity Analysis

The complete procedure of Evolving Parameter Isolation (EPI) is summarized in Algorithm 1. EPI augments standard SFT with a lightweight online sensitivity tracker and a periodically updated isolation mask. At each step, the method estimates parameter importance from gradient magnitudes, updates a binary mask every H steps to protect the most sensitive parameters, and only applies optimization updates to the remaining parameters. This design provides a simple mechanism to balance stability (preserving prior knowledge) and plasticity (adapting to new tasks) without introducing additional networks or task-specific heads.

Memory Overhead. EPI stores the sensitivity accumulator $\boldsymbol{S}_t$, which has the same dimensionality as the model parameters and therefore incurs a linear memory overhead of $O(d)$, where d is the number of parameters. In practice, this is comparable to maintaining one extra optimizer state (e.g., the second-moment estimate in Adam/AdamW). Since modern fine-tuning is typically dominated by activation memory (especially under long sequences and large batch sizes), the additional storage for $\boldsymbol{S}_t$ is minor.

Computational Efficiency. The element-wise EMA update and masking are $O(d)$ operations, incurring negligible overhead relative to the matrix multiplications in forward/backward passes. EPI supports memory sharding in distributed settings (e.g., DeepSpeed ZeRO), and since the quantile selection is triggered only every H steps, the amortized complexity remains essentially identical to standard AdamW-based SFT.

Notes on Hyperparameters. The sparsity ratio p and EMA factor β regulate protection intensity and estimate smoothness, respectively; specifically, a higher β filters stochastic noise from mini-batches. The interval H balances responsiveness with overhead, where a smaller H allows the model to adapt rapidly to task transitions, while a larger H favors stability in stationary environments.

Algorithm 1 Evolving Parameter Isolation (EPI)

Require: Pre-trained weights $\boldsymbol{\theta}_0$, Dataset $\mathcal{D}$, Sparsity ratio p, EMA factor β, Interval H, Learning rate η_t.
Ensure: Optimized parameters $\boldsymbol{\theta}_T$.

Initialize: $\boldsymbol{S}_0 \leftarrow \mathbf{0}$, $\boldsymbol{m}_0 \leftarrow \mathbf{0}$ ▷ *Initial mask: all trainable*
for $t = 1$ **to** T_{steps} **do**
  Minibatch Sampling: $(x, y) \sim \mathcal{D}$.
  Gradient Computation: $\boldsymbol{g}_t \leftarrow \nabla_{\boldsymbol{\theta}} \mathcal{L}(f_{\boldsymbol{\theta}_{t-1}}(x), y)$.
  // Phase 1: Online Sensitivity Accumulation
  $\boldsymbol{S}_t \leftarrow \beta \boldsymbol{S}_{t-1} + (1-\beta)(\boldsymbol{g}_t \odot \boldsymbol{g}_t)$
  // Phase 2: Dynamic Mask Evolution
  if $t \equiv 0 \pmod{H}$ **then**
    $\hat{\boldsymbol{I}} \leftarrow \text{LayerMinMax}(\boldsymbol{S}_t)$ ▷ *Layer-wise Normalization*
    $\tau \leftarrow \text{Quantile}(\hat{\boldsymbol{I}}, 1-p)$ ▷ *Compute dynamic threshold*
    $\boldsymbol{m}_t \leftarrow \mathbb{I}(\hat{\boldsymbol{I}} \geq \tau)$ ▷ *Update binary isolation mask*
  else
    $\boldsymbol{m}_t \leftarrow \boldsymbol{m}_{t-1}$ ▷ *Retain previous mask*
  end if
  // Phase 3: Sparse Optimization
  $\Delta\boldsymbol{\theta}_t \leftarrow \text{AdamWStep}(\boldsymbol{\theta}_{t-1}, \boldsymbol{g}_t, \eta_t)$ ▷ *Compute AdamW update*
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} + (\mathbf{1} - \boldsymbol{m}_t) \odot \Delta\boldsymbol{\theta}_t$ ▷ *Mask the update*
end for

E Additional Experimental Results

E.1 Statistical Robustness

To assess robustness, we report mean ± standard deviation over three independent runs for the main results on LLaMA-3-8B and Gemma-2-9B.

The results in Table 9 show that the gains of EPI are stable across random seeds. In particular, EPI consistently outperforms the strongest baseline, Static Isolation, on both LLaMA-3-8B and

Task	Input Prompt	Full SFT Output	Static Mask Output	EPI Output (Ours)
GSM8K (Reasoning)	A train leaves city A at 9:00 AM... (Time/Speed problem)	1:30 PM	1:35 PM	1:30 PM, with step-by-step reasoning showing distance covered by each train.
CodeAlpaca (Python)	Write a function that returns Fibonacci numbers up to n.	Returns first 5 numbers only, ignores n.	Correct function but inefficient loop, $O(n^2)$.	Correct and efficient function using iterative approach, handles all n.
Alpaca (Instruction)	Summarize the plot of 'Pride and Prejudice' in 2-3 sentences.	Misses key relationships, inconsistent with timeline.	Basic summary, minor omissions.	Concise and accurate summary including major characters, timeline, and main conflict.
UltraChat (Dialogue)	User: How do I cook pasta perfectly?	Overgeneralized advice, misses boiling details.	Correct steps but unordered.	Step-by-step guidance with tips for timing, salt, and texture; natural conversational tone.

Table 8: Qualitative comparison of model outputs. EPI demonstrates superior reasoning consistency and completeness compared to baselines, effectively mitigating the trade-off between forgetting and learning.

Base Model	Method	GSM8K	CodeAlpaca	LogiQA	Alpaca	UltraChat	Avg. Norm.
LLaMA-3-8B	Full SFT	55.6±0.4	28.3±0.5	60.8±0.5	7.8±0.2	8.1±0.1	7.32±0.06
	Static Isolation	59.2±0.6	29.4±0.4	63.0±0.4	8.1±0.1	8.3±0.1	7.68±0.04
	EPI (Ours)	**61.8±0.3**	**31.2±0.3**	**65.4±0.5**	**8.4±0.1**	**8.6±0.1**	**7.98±0.03**
Gemma-2-9B	Full SFT	58.5±0.5	30.1±0.6	64.0±0.6	8.3±0.2	8.5±0.2	7.78±0.07
	Static Isolation	62.2±0.4	31.5±0.5	66.4±0.5	8.6±0.1	8.8±0.1	8.21±0.05
	EPI (Ours)	**65.0±0.4**	**33.2±0.5**	**69.1±0.4**	**8.9±0.1**	**9.1±0.1**	**8.57±0.04**

Table 9: **Main results with standard deviations (3 runs).** EPI consistently outperforms baselines with small standard deviations across tasks, demonstrating that the gains reported in the main text are statistically robust rather than artifacts of a favorable seed.

Gemma-2-9B, with relatively small standard deviations across tasks. This suggests that the improvements reported in the main text are robust rather than artifacts of random initialization or data order.

E.2 Additional Mask Dynamics Analysis

To further quantify temporal drift, we analyze the evolution of isolation masks from three complementary perspectives: temporal overlap with the initial mask, layer-wise flip rates, and task-conditioned overlap across datasets. Together, these analyses show that mask evolution is substantial over time, structured across layers, and task-dependent across benchmarks.

Table 11 shows direct evidence of temporal drift, with initial mask overlap falling below 50% by training's end. Although Eq. 4 uses Hamming distance globally, we employ Jaccard similarity to evaluate the sparse active subsets to avoid background zero dominance. This ongoing parameter replacement explains why static masks inevitably misalign with the optimization trajectory.

Table 12 further shows that mask evolution is not uniform across the network. Upper layers exhibit substantially higher flip rates than lower layers, suggesting that higher-level reasoning and task-specific behaviors require more plasticity, whereas lower layers remain comparatively stable.

Finally, Table 13 shows that different tasks rely on only partially overlapping sets of important parameters. The relatively low Jaccard similarities confirm that heterogeneous tasks induce different critical regions, further motivating a dynamic isolation mechanism that can adapt as the task distribution changes.

E.3 Additional Analysis on Static Isolation

To complement the p-sensitivity analysis of EPI in the main text, we also evaluate static isolation under different protection ratios.

Table 10 shows that static isolation also follows an inverted-U trend with respect to the isolation ratio p. Performance peaks at $p = 1\%$ and degrades sharply at larger ratios, indicating that excessive

Isolation Ratio p	GSM8K	CodeAlpaca	LogiQA	Alpaca	UltraChat	Avg. Norm.
0.5%	58.7	28.9	62.3	8.0	8.2	7.59
1%	**59.2**	**29.4**	**63.0**	**8.1**	**8.3**	**7.68**
2%	58.4	28.5	61.8	7.8	7.9	7.42
5%	56.9	27.6	60.1	7.4	7.6	7.12

Table 10: **Static Isolation Ratio Sweep on LLaMA-3-8B**. Static isolation peaks at $p = 1\%$ and drops sharply for $p \geq 2\%$, directly supporting over-protection as a failure mode of static masks.

Training Progress	25%	50%	75%	**100% (Final)**
Mask Overlap (%)	78.3	62.1	51.4	**44.7**

Table 11: **Temporal drift in isolation masks.** The Jaccard overlap between the initial mask and later masks steadily decreases during training.

Layer Group	Average Mask Flip Rate (%)
Layers 1–8	12.3
Layers 9–16	21.7
Layers 17–24	33.5
Layers 25–32	41.2

Table 12: **Layer-wise mask evolution.** Upper layers exhibit higher flip rates, indicating stronger adaptation to evolving task demands.

Task Pair	Jaccard Similarity (Top-1%)
GSM8K vs. CodeAlpaca	0.31
GSM8K vs. LogiQA	0.42
CodeAlpaca vs. LogiQA	0.28

Table 13: **Task-conditioned overlap of important parameters.** Different tasks rely on partially distinct critical parameter subsets.

Method	$p = 0.5\%$	$p = 1\%$	$p = 2\%$	$p = 5\%$
Static Isolation	8.2	8.3	7.9	7.6
EPI	8.4	8.6	8.4	8.2

Table 14: **UltraChat performance vs. isolation ratio**. EPI maintains higher new-task plasticity at large p by releasing outdated parameters, whereas static isolation exhibits substantially larger degradation.

static locking harms adaptation. Compared with the smoother behavior of EPI in the main text, this result suggests that static masks are more vulnerable to over-protection because they cannot release outdated parameters once training progresses.

The degradation is especially visible on UltraChat, which appears late in the training sequence and therefore serves as a sensitive indicator of remaining plasticity. As shown in Table 14, EPI remains substantially more stable at larger isolation ratios, consistent with its release-and-reselect mechanism that preserves a fixed budget while reallocating protection over time.

E.4 Diagnostic Validation of the Online Importance Criterion

To directly verify that the online importance signal remains informative under temporal drift, we compare it with the true old-task sensitivity estimated through empirical perturbations. Concretely, at multiple training stages, we sample a subset of parameters (stratified across layers). We then apply small Gaussian perturbations to these parameters and compute the induced loss change ($\Delta\mathcal{L}_{\text{old}}$) on a fixed, held-out old-task batch. This yields a per-parameter empirical sensitivity signal, which we correlate with our online importance score.

Training Stage	Pearson r	Spearman ρ
Early	0.72	0.68
Middle	0.75	0.71
Late	0.73	0.69

Table 15: **Correlation between online importance and old-task sensitivity (LLaMA-3-8B).** We sample parameters across layers, estimate true old-task sensitivity via small Gaussian perturbations (measured as $\Delta\mathcal{L}_{\text{old}}$ on a held-out batch), and correlate it with our online importance score. Strong correlations persist across all training stages.

As shown in Table 15, the online importance score remains strongly correlated with old-task sensitivity throughout training. Importantly, the correlation (both Pearson r and Spearman ρ) remains stable from early to late stages. This suggests that the EMA-based importance signal successfully tracks meaningful, sensitive coordinates even under substantial parameter drift, supporting our interpretation of EPI as a practical local stability surrogate rather than a static, one-shot estimate.