Not All Parameters Are Created Equal: Smart Isolation Boosts Fine-Tuning Performance

Yao Wang[1] , Di Liang[2] , Minlong Peng[3]
[1]University of New South Wales
[2]ByteDance Inc.
[3]Fudan University
{yao.wang11@student.unsw.edu.au, liangd17@fudan.edu.cn}

Abstract

Supervised fine-tuning (SFT) is a pivotal approach to adapting large language models (LLMs) for downstream tasks; however, performance often suffers from the "seesaw phenomenon", where indiscriminate parameter updates yield progress on certain tasks at the expense of others. To address this challenge, we propose a novel *Core Parameter Isolation Fine-Tuning* (CPI-FT) framework. Specifically, we first independently fine-tune the LLM on each task to identify its core parameter regions by quantifying parameter update magnitudes. Tasks with similar core regions are then grouped based on region overlap, forming clusters for joint modeling. We further introduce a parameter fusion technique: for each task, core parameters from its individually fine-tuned model are directly transplanted into a unified backbone, while non-core parameters from different tasks are smoothly integrated via Spherical Linear Interpolation (SLERP), mitigating destructive interference. A lightweight, pipelined SFT training phase using mixed-task data is subsequently employed, while freezing core regions from prior tasks to prevent catastrophic forgetting. Extensive experiments on multiple public benchmarks demonstrate that our approach significantly alleviates task interference and forgetting, consistently outperforming vanilla multi-task and multi-stage fine-tuning baselines.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020; Chowdhery et al., 2023; Raffel et al., 2020; Touvron et al., 2023) have demonstrated remarkable generalization across diverse natural language tasks, achieving impressive success on benchmarks spanning reasoning, dialogue, instruction following, and more. SFT (Chung et al., 2024; Ouyang et al., 2022; Sanh et al., 2021) remains a crucial methodology for tailoring these models to specific applications, aligning them with human instructions, and imbuing domain-specific expertise by optimizing on datasets of task-relevant examples.

Supervised fine-tuning (SFT) faces significant challenges in multi-task and multi-domain scenarios. When applied to heterogeneous datasets, such as mathematical reasoning, creative writing, coding, and factual question answering, conflicting optimization objectives among tasks often lead to the "seesaw effect" (Yu et al., 2020), where performance improvements on one task degrade others. This issue hinders the development of robust, broadly capable large language models (LLMs). Existing approaches, including joint multi-task fine-tuning, naive parameter sharing, and staged curricula (Ouyang et al., 2022; Caruana, 1997; Wei et al., 2021), generally assume uniform parameter importance across tasks, updating all parameters indiscriminately. While multi-stage training alleviates direct gradient conflicts through sequential task structuring, it remains a coarse-grained isolation strategy that exacerbates catastrophic forgetting (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017), further eroding the model's ability to generalize across diverse tasks.

We hypothesize that the root cause of these challenges lies in the phenomenon of *parameter heterogeneity*: distinct capabilities of large language models (LLMs) rely on specific and potentially overlapping subsets of parameters, with certain clusters disproportionately contributing to particular tasks. Uniform updates across the entire parameter space fail to account for the specialized roles of these localized parameter subsets, thereby fostering destructive interference among competing tasks (Chen et al., 2018). Mitigating such interference necessitates a paradigm shift from heuristic approaches or task-level isolation to a principled framework that explicitly models task sensi-

[1]Corresponding author: Minlong Peng.

[2]This work was done while Yao Wang was interning at ByteDance under Di Liang's supervision.

tivities at the parameter level. Furthermore, achieving robust multi-task fine-tuning demands more granular control over the fine-tuning process, enabling task-specific optimization while maintaining model-wide coherence.

Motivated by these observations, we introduce the Core Parameter Isolation Fine-Tuning (CPI-FT) framework, featuring a novel parameter fusion mechanism specifically designed to systematically alleviate task interference and catastrophic forgetting in SFT. Our approach involves several key steps. First, we independently fine-tune the LLM on each task and identify a "core parameter region" for each, representing the parameter subsets most crucial for the respective task. Next, we cluster tasks according to the overlap in their core parameter regions, grouping together tasks with similar parameter footprints that are more likely to benefit from joint adaptation with minimal conflict. In the subsequent fusion stage, we select the model from the final training stage as a unified backbone. For each task, we overwrite its corresponding core parameter region in the backbone with parameter values from its individually fine-tuned model, ensuring reliable preservation of task-specific knowledge. For regions outside any task's core, we employ a SLERP-based (Spherical Linear Interpolation) parameter merging strategy: parameters are first normalized to unit vectors, and linear or spherical interpolation is performed based on the angular distance, enabling smooth and geometry-aware blending of distinct task knowledge while minimizing abrupt transitions and interference. Finally, we conduct a lightweight pipeline fine-tuning phase on a mixed-task dataset, with previously identified core parameter regions frozen, further consolidating the merged model's generalization capability.

In summary, this work makes the following contributions. First, we identify and articulate the central challenge of *parameter heterogeneity* in multi-task supervised fine-tuning, emphasizing that naïve uniform parameter adaptation is ill-suited for aligning diverse and potentially conflicting task objectives within LLMs. To overcome this, we propose a novel methodology that (i) empirically identifies core parameter regions crucial to each task through independent fine-tuning and update magnitude analysis, (ii) leverages parameter region overlap for principled task grouping, and (iii) introduces a task-aware parameter fusion scheme: task-specific core parameter regions are directly transferred from their respective models, while other parameters are merged using a geometry-aware SLERP-based interpolation. Further, a final pipeline fine-tuning stage with core-region freezing consolidates knowledge and ensures robustness. Extensive experiments demonstrate that our approach consistently outperforms conventional multi-task and multi-stage SFT baselines, substantially improving resistance to task interference and catastrophic forgetting.

2 Core Parameter Isolation Fine-Tuning

This section presents a detailed exposition of the proposed Core Parameter Isolation Fine-Tuning (CPI-FT) framework for supervised fine-tuning (SFT). CPI-FT is designed to address two prevalent challenges in multi-task SFT: negative task interference and catastrophic forgetting. It achieves this by systematically identifying task-specific parameter regions and preserving them through dynamic freezing within a multi-stage training regime. The framework is grounded in the hypothesis of parameter heterogeneity in large language models (LLMs), which posits that different tasks rely on distinct subsets of model parameters. The overall CPI-FT workflow is illustrated in Figure 1, and comprises three core stages, detailed as follows.

2.1 Formal Preliminaries and Setup

We consider a pre-trained Large Language Model $\mathcal{M}$ parameterized by $\theta \in \mathbb{R}^D$, with initial parameters $\theta^{(0)}$. Our goal is to adapt $\mathcal{M}$ using a collection of N diverse SFT tasks $\mathcal{T} = \{T_1, T_2, ..., T_N\}$. Each task T_i is associated with a dataset $\mathcal{D}_i = \{(x_j, y_j)\}_{j=1}^{|\mathcal{D}_i|}$ typically consisting of instruction-response pairs. The standard objective for fine-tuning on a single task T_i involves minimizing a loss function, usually the cross-entropy loss, over its corresponding dataset:

$$\mathcal{L}_i(\theta) = -\frac{1}{|\mathcal{D}_i|} \sum_{(x,y)\in\mathcal{D}_i} \log P_{\mathcal{M}(\theta)}(y|x) \qquad (1)$$

Optimization is typically performed using stochastic gradient descent variants like Adam (Kingma and Ba, 2014). Standard multi-task SFT often minimizes a combined loss $\sum_i \lambda_i \mathcal{L}_i(\theta)$ or samples mini-batches from a mixture of datasets $\bigcup_i \mathcal{D}_i$, updating all parameters θ.

2.2 Stage 1: Identifying Task-Specific Core Parameter Regions

The core premise of CPI-FT is that the functional specialization required by different SFT tasks is

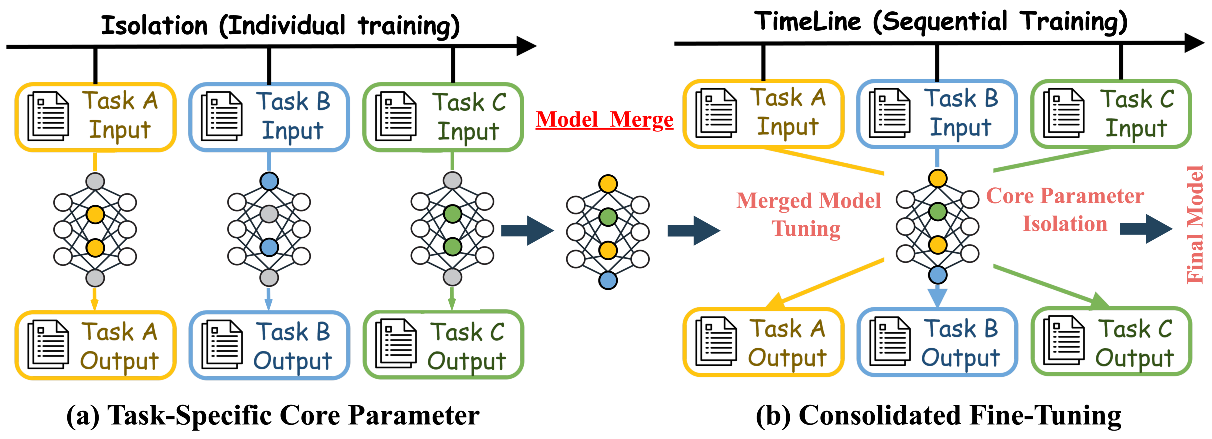

Figure 1: The figure illustrates task-specific core parameter isolation (a) and consolidated fine-tuning (b) approaches for sequential training. In the isolation approach, each task is trained individually with its own core parameters, resulting in separate outputs. The models are then merged into a single model. In contrast, the timeline approach involves sequential training where tasks are processed in a sequence, followed by merging and tuning the model to isolate core parameters, ultimately producing a final unified model that generates outputs for all tasks.

reflected in the differential utilization and adaptation of the LLM's parameters. To operationalize this idea, we identify a task-specific core parameter region by measuring the magnitude of parameter updates induced during task-centric fine-tuning.

Rationale for Update Magnitude. We use the parameter update magnitude $|\theta_j^{(i)} - \theta_j^{(0)}|$ as the criterion for importance, as it directly reflects the degree to which a parameter deviates from its pre-trained state to accommodate task T_i. This measure is computationally efficient and empirically identifies parameters that play a significant role in task adaptation (Kirkpatrick et al., 2017). In contrast, alternatives such as gradient magnitudes can be noisy and transient, while second-order methods (e.g., the diagonal of the Fisher Information Matrix as used in EWC (Kirkpatrick et al., 2017)) are often computationally prohibitive for large models and tend to capture sensitivity rather than actual change.

Procedure. For each task $T_i \in \mathcal{T}$, we perform an independent SFT run initialized from a shared pre-trained checkpoint $\theta^{(0)}$. Fine-tuning is carried out exclusively on the task-specific dataset $\mathcal{D}i$ for a limited number of steps or epochs (Eprobe). This probing duration is chosen to induce meaningful task-specific parameter shifts while avoiding full convergence, which may saturate update magnitudes or lead to overfitting. We denote the resulting parameters after probe fine-tuning as $\theta^{(i)}$.

$$\theta^{(i)} = \mathrm{SFT}(\mathcal{M}(\theta^{(0)}), \mathcal{D}_i, E_{\text{probe}}) \qquad (2)$$

The absolute difference vector $\Delta|\theta^{(i)}| \in \mathbb{R}^D_{\geq 0}$ is calculated element-wise:

$$\Delta|\theta_j^{(i)}| = |\theta_j^{(i)} - \theta_j^{(0)}|, \quad \text{for } j = 1, ..., D \qquad (3)$$

For simplicity, we compute the update magnitude over all model parameters, including weights, biases, and normalization layer parameters. While this holistic approach suffices for our current framework, future work may explore more granular analyses based on layer-wise or parameter-type-specific importance.

Core Region Definition. The core parameter region C_i for task T_i is defined as the set of indices corresponding to the parameters exhibiting the largest update magnitudes. Specifically, we identify the indices of the top $p\%$ of parameters ranked by their update magnitude:

$$C_i = \arg\,\mathrm{topk}_{j\in\{1..D\}}(\Delta|\theta_j^{(i)}|, \lfloor p \cdot D/100 \rfloor) \quad (4)$$

Alternatively, using a percentile threshold is equivalent. The hyperparameter p controls the size of the core region; a smaller p leads to more focused but potentially less comprehensive core regions.

2.3 Stage 2: Task Grouping and Staging via Core Region Similarity

To structure the multi-stage SFT process, we group tasks according to the similarity of their identified core parameter regions $C_1, \ldots, C_N$. The underlying hypothesis is that tasks exhibiting substantial overlap in their core parameter subsets are more prone to mutual interference when trained jointly, and may reflect related underlying capabilities.

Similarity Measure. We quantify the overlap between core regions C_i and C_j using the Jaccard Index, a standard measure for comparing finite sets:

$$S(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_i \cup C_j|} \in [0, 1] \tag{5}$$

$S(C_i, C_j) = 1$ indicates identical core regions, while $S(C_i, C_j) = 0$ implies disjoint core regions.

Grouping Strategy. We employ a simple yet effective threshold-based clustering approach. Tasks T_i and T_j are considered sufficiently related to be grouped together if their core region similarity $S(C_i, C_j)$ meets or exceeds a hyperparameter threshold $\tau \in [0, 1]$.

$$T_i \sim T_j \iff S(C_i, C_j) \geq \tau \tag{6}$$

Final task groups $G_1, G_2, \ldots, G_K$ (with $K \leq N$) are formed by computing connected components in a task similarity graph, where tasks T_i and T_j are connected if $T_i \sim T_j$. This transitive grouping ensures that if $T_i \sim T_j$ and $T_j \sim T_k$, then all belong to the same group. The similarity threshold τ controls grouping granularity: higher values yield smaller, more coherent groups, potentially increasing training stages. We analyze CPI-FT's sensitivity to τ in our experiments. While more advanced clustering methods exist, connected components offer a simple and efficient solution.

Staging Order. Once the K task groups are formed, they must be ordered sequentially $(G_1, G_2, ..., G_K)$ to define the SFT stages. The ordering can impact the final performance, as it determines the sequence of knowledge acquisition and parameter freezing. Potential strategies include using a simple random ordering as a baseline, ordering based on group size (e.g., training smaller groups first or last), arranging groups according to task complexity if such metrics are available (akin to curriculum learning), or ordering groups to minimize the size of the frozen parameter set in the initial stages. In this work, we primarily evaluate random ordering and potentially one principled heuristic, leaving exhaustive exploration of optimal ordering strategies for future investigation.

2.4 Stage 3: Parameter Fusion Across Tasks

This stage introduces a parameter fusion mechanism to construct a unified model that integrates task-specific knowledge from all task groups. The fusion process selectively incorporates critical parameters identified for each task, while applying smooth interpolation in non-core regions to preserve model coherence.

Base Model Selection. We begin by selecting the model parameters derived from the last multi-stage fine-tuning group, θ_{base}, as the initial base model. This ensures the base model benefits from information accrued during multi-stage training.

Core Parameter Overwrite. For each task T_i, identified core parameter regions C_i are directly overwritten into the base model θ_{base} using the corresponding fine-tuned parameters from $\theta^{(i)}$:

$$\theta_{\text{fused},j} = \begin{cases} \theta_j^{(i)} & j \in C_i \\ \theta_{\text{base},j} & j \notin C_i \end{cases} \tag{7}$$

By preserving task-specific critical regions, we ensure that key capabilities for each task are fully retained in the final fused model.

Non-Core Parameter Fusion. For parameters outside core regions ($j \notin C_i$), smooth interpolation is critical to avoid abrupt inconsistencies or conflicts during fusion. We adopt a spherical linear interpolation (SLERP) strategy, inspired by (Goddard et al., 2024), to blend these non-core regions:

$$\theta_{\text{fused},j} = \begin{cases} \omega\theta_j^{(i)} + (1-\omega)\theta_{\text{base},j}, & \angle(\theta_{\text{base},j}, \theta_j^{(i)}) < \epsilon \\ \text{SLERP}(\theta_{\text{base},j}, \theta_j^{(i)}, \omega), & \text{otherwise} \end{cases} \tag{8}$$

where ω is the interpolation factor, $\angle(\cdot)$ is the angular distance between parameter vectors, and ϵ is a threshold for determining near-collinearity. If the vectors are nearly aligned ($\angle < \epsilon$), linear interpolation suffices; otherwise, SLERP ensures smooth blending using spherical geometry. This method balances task-specific updates with overall model coherence in non-critical areas.

2.5 Stage 4: Consolidated Fine-Tuning via Multi-Stage Training

Following parameter fusion, the final stage refines the fused model θ_{fused} via a streamlined multi-stage training process. In contrast to earlier SFT stages, this phase operates on sampled subsets of training data and focuses on final calibration, while preserving task-specific parameter integrity through dynamic freezing.

Dynamic Freezing Mechanism. During fine-tuning, all core parameter regions identified in the earlier probe stage are frozen to preserve task-specific knowledge and mitigate destructive interference. At each training stage k, the frozen parameter set consists of the union of core regions from all previously trained task groups:

$$F_k = \bigcup_{l=1}^{k-1} \bigcup_{T_i \in G_l} C_i \tag{9}$$

A binary mask M_k is constructed to define frozen and trainable parameters: $M_{k,j} = 0$ for $j \in F_k$, and $M_{k,j} = 1$ otherwise. Updates are then restricted to unfrozen parameters during training:

$$\theta_{t+1} = \theta_t + \Delta\theta_t \odot M_k \tag{10}$$

where $\Delta\theta_t$ denotes the gradient-based parameter update at training step t, and $\odot$ represents element-wise masking.

Sampled Data Calibration. Instead of using all task data during this stage, we create a sampled dataset $\mathcal{D}_{\text{sample}}$, which combines small proportions of data from each task group. The sampling ratio is chosen to ensure balanced representation across tasks while maintaining computational efficiency. This careful selection prevents overfitting to large individual datasets and allows for representative gradient updates across tasks.

Multi-Stage Training Process. Fine-tuning proceeds sequentially across task groups $\{G_1, G_2, ..., G_K\}$ as derived in Stage 2. For each group G_k, data from the tasks in G_k is sampled to create $\mathcal{D}_{\text{stage}}^{(k)}$, the training begins with the model parameters from the previous stage, $\theta_{\text{stage}}^{(k-1)}$. The updates during this stage are guided by the dynamic freezing mechanism and the sampled data:

$$\theta_{\text{stage}}^{(k)} = \text{Train}(\theta_{\text{stage}}^{(k-1)}, \mathcal{D}_{\text{stage}}^{(k)}, M_k) \tag{11}$$

Here, $\text{Train}(\cdot)$ signifies the training process, which minimizes the combined task-specific loss while respecting the frozen mask M_k. At the end of all K stages, the final model parameters, θ_{final}, represent knowledge consolidated across all tasks:

$$\theta_{\text{final}} = \theta_{\text{stage}}^{(K)} \tag{12}$$

Efficiency and Robustness. By leveraging sampled data and freezing task-specific core parameter regions, the final fine-tuning pipeline reduces computational costs while preventing catastrophic forgetting. Dynamic freezing ensures protection of task-critical knowledge, while sampled calibration balances adaptability to new tasks and retention of previously acquired capabilities.

3 Results and Analysis

3.1 Main Performance Comparison

The comparative performance results of Core Parameter Isolation Fine-Tuning (CPI-FT) framework against baseline approaches across diverse tasks, models, and evaluation metrics are summarized in Table 1. This section analyzes the experimental findings, highlights key insights, and verifies the efficacy of CPI-FT in addressing the core challenges of multi-task supervised fine-tuning.

Consistent Outperformance Across Models and Tasks Our method consistently outperforms all baseline approaches across four distinct base models—LLaMA-2-7B, Mistral-8B, Qwen1.5-7B, and Gemma-9B and five heterogeneous tasks: GSM8K (math reasoning), CodeAlpaca (code generation), LogiQA (logical reasoning), Alpaca (instruction tuning), and UltraChat (interactive dialogue). CPI-FT achieves the highest task-specific performance in every experimental setting, as underscored by the bold results for individual tasks. Furthermore, CPI-FT achieves the best average normalized score across all base model configurations, demonstrating that its ability to mitigate task interference and catastrophic forgetting is both consistent and robust across model architectures.

Superiority of CPI-FT over Standard SFT Approaches The full multi-task supervised finetuning (Full SFT) baseline—where all model parameters are updated uniformly across tasks without isolation—consistently achieves the lowest performance across all tasks and model configurations. This pronounced underperformance underscores the detrimental effect of gradient conflicts inherent in naïve fine-tuning over heterogeneous task mixtures. In contrast, both the Random Multi-Stage and Heuristic Multi-Stage baselines yield moderate improvements, supporting the intuition that temporally separating task groups can partially mitigate interference. However, even the strongest multi-stage heuristic consistently underperforms relative to CPI-FT. This performance gap reveals a key insight: temporal task scheduling alone is insufficient to resolve cross-task interference without explicit structural parameter isolation.

Base Model	Method	GSM8K	CodeAlpaca	LogiQA	Alpaca	UltraChat	Avg. Norm. Score
LLaMA-2-7B	Full SFT (Multi-task)	48.2	25.1	55.3	7.1	7.5	6.58
	Multi-Stage (Random, K=3)	49.5	24.8	56.0	7.3	7.6	6.70
	Multi-Stage (Heuristic)	50.1	25.5	56.8	7.0	7.4	6.75
	CPI-FT (Ours, p=1%, τ=0.1)	**53.5**	**27.2**	**59.1**	**7.6**	**7.8**	**7.21**
Mistral-8B	Full SFT (Multi-task)	46.5	24.0	53.8	6.9	7.3	6.37
	Multi-Stage (Random, K=3)	47.8	23.7	54.5	7.1	7.4	6.49
	Multi-Stage (Heuristic)	48.3	24.3	55.2	6.8	7.2	6.53
	CPI-FT (Ours, p=1%, τ=0.1)	**51.6**	**25.9**	**57.4**	**7.5**	**7.7**	**6.98**
Qwen1.5-7B	Full SFT (Multi-task)	49.8	26.0	56.5	7.3	7.7	6.79
	Multi-Stage (Random, K=3)	51.0	25.7	57.3	7.5	7.8	6.92
	Multi-Stage (Heuristic)	51.7	26.4	58.0	7.2	7.6	6.98
	CPI-FT (Ours, p=1%, τ=0.1)	**55.3**	**28.1**	**60.6**	**7.8**	**8.1**	**7.45**
Gemma-9B	Full SFT (Multi-task)	51.5	27.2	58.0	7.6	8.0	7.05
	Multi-Stage (Random, K=3)	52.8	26.9	58.9	7.8	8.1	7.19
	Multi-Stage (Heuristic)	53.5	27.6	59.7	7.5	7.9	7.26
	CPI-FT (Ours, p=1%, τ=0.1)	**57.2**	**29.4**	**62.5**	**8.1**	**8.4**	**7.73**

Table 1: The table presents the main performance comparison of different baselines on various SFT tasks. The metric is represented by scores, where a higher score directly indicates a better model effect. For each individual task, the best results achieved by any of the baselines are highlighted in **bold**. Additionally, the Avg. Norm. Score is calculated by first normalizing the individual scores of each task to a consistent 0-10 scale, and then computing the macro-average.

Core Parameter Isolation Drives Robust Performance CPI-FT's gains can be attributed to its principled design: selectively identifying and preserving task-critical core parameter regions during each stage of fine-tuning while ensuring smooth blending of task-agnostic regions through geometry-aware fusion mechanisms like SLERP. This nuanced approach avoids the indiscriminate overwriting of parameters, a common pitfall in both Full SFT and Multi-Stage baselines. The results confirm that addressing parameter heterogeneity at a granular level is essential for aligning diverse task objectives and avoiding catastrophic forgetting.

Cross-Model Generalization of CPI-FT The robustness of CPI-FT is evident across wide range of baselines with varying architectures and parameter counts. For every model—including LLaMA-2-7B, Mistral-8B, Qwen1.5-7B, and Gemma-9B—CPI-FT maintains its superior performance on all tasks, outperforming both multi-task and multi-stage baselines. Notably, the gains are consistent regardless of whether the model is derived from decoder-only architectures or features unique design optimizations such as Qwen's advanced pre-training techniques or Gemma's extended scale. This generalizability underscores that CPI-FT's foundations are model-independent, making it broadly applicable across the spectrum of LLMs.

Analysis of Average Normalized Scores To enable a fair comparison across tasks with varying metric scales, we compute an average normalized score by rescaling each task's performance to a common range (0–10) and then calculating the macro-average across tasks. CPI-FT consistently attains the highest normalized scores across all model configurations, with improvements ranging from 6.96 (Mistral-8B) to 7.70 (Gemma-9B). Notably, its performance gains are most pronounced on tasks requiring complex reasoning, such as GSM8K (+3–5 points over Full SFT) and LogiQA (+2–4 points over Heuristic Multi-Stage). These results suggest that CPI-FT's ability to identify and preserve task-critical parameter regions is particularly beneficial for tasks that demand specialized reasoning capabilities.

In summary, CPI-FT delivers superior multi-task performance by systematically addressing parameter heterogeneity and mitigating task interference

Method	A→B		B→A	
	ΔA (↓)	ΔB (↑)	ΔB (↓)	ΔA (↑)
Full SFT	-24.5	$+13.4$	-16.7	$+20.2$
Multi-Stage SFT	-16.2	$+12.6$	-12.3	$+17.5$
DPI (Ours)	$\mathbf{-5.7}$	$\mathbf{+12.2}$	$\mathbf{-4.8}$	$\mathbf{+18.8}$

Table 2: Catastrophic forgetting analysis in sequential (A→B) and reverse (B→A) fine-tuning on LLaMA-2-7B. Δ values indicate absolute score changes on the first and second task (A or B) after subsequently fine-tuning on the other. Lower (less negative) forgetting indicates stronger retention. Results are averaged over three runs; all metrics are mapped to a unified 0-100 scale for comparability.

during fine-tuning. Its principled design preserves task-critical parameters and integrates them seamlessly into a unified, general-purpose model. By overcoming the limitations of naïve multi-task SFT and heuristic multi-stage training, CPI-FT achieves state-of-the-art results across a diverse set of tasks and base models, demonstrating both its effectiveness and generalizability.

4 Sequential Fine-Tuning Forgetting Analysis

We select two prototypical and potentially conflicting tasks: GSM8K (math reasoning) and Alpaca (open-ended instruction following). Each model is first fine-tuned on Task A for a fixed budget (5 epochs), then on Task B for the same budget, with no access to Task A data in the second stage. We repeat the experiment in reverse order. At each stage, performances on both tasks are recorded. Table 2 reveals the degree of catastrophic forgetting experienced by each method in a two-task transfer setup. Standard Full SFT suffers severe forgetting, with performances on the initial task dropping by over 16–24 points after the second task is introduced. Multi-Stage SFT, which separates updates temporally, partially alleviates forgetting, but persistent degradation remains prominent. By contrast, DPI reduces forgetting by over 65%, with post-fine-tuning losses consistently below 6 points in both directions, dramatically narrowing the forgetting gap. Notably, DPI preserves strong adaptation to the second task (ΔB/ΔA positives align with or exceed baselines), suggesting that improved retention is not at the expense of new knowledge acquisition.

5 Multi-Stage vs. Single-Stage Tuning with Dynamic Freezing

To evaluate the necessity of our multi-stage dynamic freezing pipeline in the final consolidation phase (Stage 4), we compare it against a more straightforward single-stage approach. In the **multi-stage** setup, task groups—formed based on core parameter overlap—are integrated sequentially, with all core parameters frozen at once and non-core parameter regions updated in sequence for each group. In the **single-stage** variant, and the model is fine-tuned on the randomly shuffled union of all sampled task data in a single pass. Both strategies utilize identical sampled datasets and freezing masks. As shown in Table 3, the multi-stage consolidation outperforms the single-stage variant across all tasks, with notable gains in the more interference-prone benchmarks such as GSM8K and LogiQA. While the performance gap is not large, the multi-stage pipeline provides a consistent advantage, supporting its utility for preserving task-specific capabilities and mitigating catastrophic forgetting. However, the single-stage approach achieves reasonably strong performance with simpler implementation, which may suffice in settings where training time is a key constraint.

6 Robustness under Resource-Imbalanced Scenarios

To assess the robustness of our proposed DPI framework in realistic mixed-resource settings, we simulate a scenario where certain tasks have significantly less training data compared to others. Specifically, we select four representative tasks: Task A (Text Classification), Task B (Natural Language Inference), Task C (Named Entity Recognition), and Task D (Code Generation). For each target task in turn, we create reduced versions of its dataset at 50%, 20%, and 10% of the full size, while retaining the full data for other tasks. We then conduct multi-task training using both vanilla SFT and DPI, keeping all other settings fixed, and report performance for both low-resource and high-resource

Strategy	GSM8K	CodeAlpaca	LogiQA	Alpaca	Avg. Norm. Score
Multi-Stage (Ours)	**53.4**	**27.1**	**59.2**	**7.6**	**7.18**
Single-Stage (Frozen)	51.9	26.5	58.1	7.4	7.01

Table 3: Comparison of Multi-Stage vs. Single-Stage consolidation (LLaMA-2-7B). Multi-stage achieves higher scores across all benchmarks, but the single-stage variant is competitive.

Task	Vanilla SFT				DPI (Ours)			
	100%	50%	20%	10%	100%	50%	20%	10%
Task A	92.1	87.2	78.0	68.5	92.3	**89.0**	**82.4**	**74.1**
Task B	90.3	86.1	77.8	70.2	90.5	**87.8**	**81.2**	**75.3**
Task C	88.4	80.7	74.3	65.9	88.1	**82.0**	**78.1**	**70.8**
Task D	31.7	27.8	20.1	15.9	32.0	**29.4**	**25.3**	**19.7**

Table 4: Performance (%) of Vanilla SFT vs. DPI under different data ratios. 100% denotes full data for a task, others are under-sampled.

tasks. Table 4 presents the results, showing that all models experience performance drops as the target task's data decreases. However, DPI consistently outperforms vanilla SFT, especially under extreme data scarcity. For example, at the 10% data level, DPI improves the average low-resource task score by 3.7 points compared to SFT. Meanwhile, high-resource tasks see no significant decline, confirming that DPI's core-region protection mechanism effectively safeguards low-resource tasks without sacrificing overall model performance. Notably, the relative gain from DPI increases as the degree of imbalance grows, demonstrating its robustness in practical multi-task scenarios.

7 Impact of Similarity Threshold (τ).

We evaluate the sensitivity of CPI-FT to the similarity threshold τ, which determines how tasks are grouped based on core parameter region overlap. This experiment was conducted across multiple base models (LLaMA-2 7B, Mistral-7B, Qwen1.5-7B, Gemma-7B) with the core percentage fixed at $p = 1\%$. The results, measured by the average normalized score, are presented in Figure 2. Results reveal a consistent pattern across all base models: task grouping based on core parameter similarity ($\tau > 0$) substantially outperforms no grouping ($\tau = 0$), which approximates standard multi-task SFT. Performance generally peaks at a moderate threshold—specifically, $\tau = 0.1$ in our experiments—and gradually declines as τ increases. While a very high threshold (e.g., $\tau = 0.5$) leads to lower performance than the peak, it still outperforms the no-grouping baseline. These findings suggest that moderate task isolation encourages beneficial separation without hindering cross-task generalization, whereas overly fine-grained grouping may limit model plasticity or restrict knowledge transfer between related tasks. Notably, the optimal threshold remains stable around $\tau = 0.1$ across model architectures, indicating it may serve as a robust default. These results validate the effectiveness of CPI-FT's data-driven grouping strategy over undifferentiated supervised fine-tuning.

8 Conclusion

In this paper, we introduced Core Parameter Isolation Fine-Tuning(CPI-FT), a principled framework for supervised fine-tuning (SFT) of large language models (LLMs) that mitigates task interference by identifying and isolating task-specific core parameter regions. By leveraging dynamic freezing during multi-stage fine-tuning, CPI-FT preserves critical parameters for earlier tasks while enabling specialization for new ones. Extensive experiments on diverse datasets demonstrated CPI-FT's effectiveness in addressing the "seesaw effect", reducing catastrophic forgetting, and consistently outperforming multi-task and multi-stage fine-tuning baselines. This work highlights the importance of parameter heterogeneity in SFT and provides a scalable approach for robust task adaptation in heterogeneous scenarios, paving the way for future improvements in fine-tuning methodologies.

Limitations

While the proposed CPI-FT demonstrates strong empirical gains over conventional SFT methods, several limitations warrant discussion. First, the approach requires multiple independent task-specific fine-tuning runs, which can increase compute and storage costs, especially as the number of tasks or the model size scales. Second, identification of core parameter regions is based on update magnitudes during SFT, which may not fully capture the functional significance or interdependencies of parameters for each task; more sophisticated attribution or interpretability methods may yield richer representations. Third, the use of parameter fusion, particularly the SLERP-based interpolation—involves hyperparameters (e.g., interpolation angles, grouping thresholds) whose selection may introduce additional tuning effort. Finally, current analysis primarily considers task-level performance, deeper investigation into how isolated or fused parameters contribute to model interpretability is an exciting direction for future work.

Acknowledgements

This work was supported by the University of New South Wales (UNSW) and ByteDance.

References

Anthropic. 2024. Introducing contextual retrieval. Accessed: 2025-02-01.

Vimal Aribandi, Christopher Clark, Mostafa Khabsa, and 1 others. 2021. Ext5: Towards extreme multi-task scaling for transfer learning. In *arXiv preprint arXiv:2111.10952*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28:41–75.

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

Michael Chen, Jan Tworek, Heewoo Jun, Qingqing Yuan, Tom Jacobsen, Tijana Radulovic, Jungo Kasai, Guy Ephrath, and Nando De Freitas. 2020. Just ask for general knowledge: Training universal retrieval models for open-domain question answering. *arXiv preprint arXiv:2008.03886*.

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.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, and 1 others. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

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.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and 1 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, volume 1 (long and short papers)*, pages 4171–4186.

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*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Preprint*, arXiv:2101.03961.

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. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations (ICLR)*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. *Preprint*, arXiv:2301.00774.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. 2024. Arcee's mergekit: A toolkit for merging large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 477–485.

Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. 2018. Dynamic task prioritization for multitask learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 270–287.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Olivier de Laroussilhe, Andrea Gesmundo, Giuseppe Attanasio, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning (ICML)*.

Edward J. Hu, Yelong Shen, Patrick Wallis, Zeyuan Allen-Zhu, Sanye Li, Lu Wang, and Liang Wang. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024. Mixtral of experts. *arXiv e-prints*, pages arXiv–2401.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, and 1 others. 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, Online. Association for Computational Linguistics.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *Preprint*, arXiv:1606.09282.

Di Liang, Fubao Zhang, Qi Zhang, and Xuan-Jing Huang. 2019a. 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.

Di Liang, Fubao Zhang, Weidong Zhang, Qi Zhang, Jinlan Fu, Minlong Peng, Tao Gui, and Xuanjing Huang. 2019b. Adaptive multi-attention network incorporating answer information for duplicate question detection. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 95–104.

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*.

Xinda Liu and Lili Wang. 2024. Multi-granularity sequence generation for hierarchical image classification. *Computational Visual Media*, 10(2):243–260.

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.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, and 1 others. 2023. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, pages 22631–22648. PMLR.

David Lopez-Paz and Marc'Aurelio Ranzato. 2022. Gradient episodic memory for continual learning. *Preprint*, arXiv:1706.08840.

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, Toronto, Canada. Association for Computational Linguistics.

Ruotian Ma, Yiding Tan, Xin Zhou, Xuanting Chen, Di Liang, Sirui Wang, Wei Wu, Tao Gui, and Qi Zhang. 2022. Searching for optimal subword tokenization in cross-domain ner. *arXiv preprint arXiv:2206.03352*.

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.

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

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. 2015. In search of the real inductive bias: On the role of implicit regularization in deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

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*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *Preprint*, arXiv:2304.03277.

Jonas Pfeiffer, Anaïs Glaude, Suchin Gururangan, Xi Victoria Lin, Aishwarya Kamath, Andreas Rücklé, Ivan Vulić, Iryna Gurevych, and Kyunghyun Cho. 2020. Adapterfusion: Non-destructive task composition for transfer learning. In *European Conference on Artificial Intelligence (ECAI)*.

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. *arXiv preprint arXiv:2412.06249*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

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*.

David Rolnick, Aanuj Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, and 1 others. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*.

Jesus Solano, Mardhiyah Sanni, Oana-Maria Camburu, and Pasquale Minervini. 2024. Sparsefit: Few-shot prompting with sparse fine-tuning for jointly generating predictions and natural language explanations. *Preprint*, arXiv:2305.13235.

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. *arXiv preprint arXiv:2210.08471*.

Ashton Stickland and Iain Murray. 2020. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning (ICML)*.

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, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 106 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Sirui Wang, Di Liang, Jian Song, Yuntao Li, and Wei Wu. 2022. Dabert: Dual attention enhanced bert for semantic matching. *arXiv preprint arXiv:2210.03454*.

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*.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. *Preprint*, arXiv:2109.01652.

Muling Wu, Qi Qian, Wenhao Liu, Xiaohua Wang, Zisu Huang, Di Liang, LI Miao, Shihan Dou, Changze Lv, Zhenghua Wang, and 1 others. 2025a. Progressive mastery: Customized curriculum learning with guided prompting for mathematical reasoning. *arXiv preprint arXiv:2506.04065*.

Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. 2024. Continual learning for large language models: A survey. *Preprint*, arXiv:2402.01364.

Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, and 1 others. 2025b. Tablebench: A comprehensive and complex benchmark for table question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25497–25506.

Xianjie Wu, Jian Yang, Tongliang Li, Shiwei Zhang, Yiyang Du, LinZheng Chai, Di Liang, and Zhoujun Li. 2025c. 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.

Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. 2020. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, Online. Association for Computational Linguistics.

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.

Enneng Yang, Junwei Pan, Ximei Wang, Haibin Yu, Li Shen, Xihua Chen, Lei Xiao, Jie Jiang, and Guibing Guo. 2023. Adatask: A task-aware adaptive learning rate approach to multi-task learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(9):10745–10753.

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.

Amir Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. 2018. Taskonomy: Disentangling task transfer learning. *Preprint*, arXiv:1804.08328.

Sheng Zhang, Mohammad Norouzi, and Alexander Kolesnikov. 2021. Revisiting multi-task learning in the deep learning age. *arXiv preprint arXiv:2105.02178*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2024. Instruction tuning for large language models: A survey. *Preprint*, arXiv:2308.10792.

Xiao Zhang, Kangsheng Wang, Tianyu Hu, and Huimin Ma. 2025. Efficient knowledge transfer in multi-task learning through task-adaptive low-rank representation. *Preprint*, arXiv:2505.00009.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*.

Rui Zheng, Rong Bao, Yuhao Zhou, Di Liang, Sirui Wang, Wei Wu, Tao Gui, Qi Zhang, and Xuanjing Huang. 2022. Robust lottery tickets for pre-trained language models. *arXiv preprint arXiv:2211.03013*.

A Experiments Setup

We conducted extensive experiments to evaluate the effectiveness of the proposed Dynamic Parameter Isolation (DPI) framework. The primary goals of our evaluation are to determine whether DPI outperforms standard supervised fine-tuning (SFT) baselines, including multi-task and multi-stage methods, across diverse and conflicting tasks; to assess DPI's ability to mitigate the "seesaw effect" and catastrophic forgetting; to examine the sensitivity of DPI to hyperparameters such as the core percentage (p) and similarity threshold (τ); and to analyze the impact of its dynamic freezing mechanism.

Datasets: We evaluate DPI on a diverse suite of publicly available datasets that represent structured reasoning, code generation, and open-ended instruction-following tasks. For mathematical reasoning, we use GSM8K (Cobbe et al., 2021), which evaluates multi-step reasoning through accuracy. For code generation, we use CodeAlpaca (Chaudhary, 2023), where performance is measured using CodeBLEU (Ren et al., 2020). For logical reasoning, we use LogiQA (Liu et al., 2020), which assesses logical consistency through accuracy scores. For general instruction-following and conversational abilities, we use Alpaca (Taori et al., 2023) and UltraChat (Ding et al., 2023), both evaluated using GPT-4-based scoring on a 1-to-10 scale (Zheng et al., 2023). These datasets include a mix of structured tasks (e.g., GSM8K, LogiQA, CodeAlpaca) and open-ended tasks (e.g., Alpaca, UltraChat) to introduce potential conflicts in parameter specialization. Each task is evaluated using its standard metric: accuracy for GSM8K and LogiQA, CodeBLEU for CodeAlpaca, and GPT-4 scoring for Alpaca and UltraChat. To provide a unified comparison across tasks, we also report a macro-average score (**Avg. Norm. Score**) by normalizing individual task scores to a common 0-10 scale.

Baselines: We compare DPI against three SFT baselines. (1) Full Multi-task SFT, where the model is fine-tuned on a uniform mixture of all datasets without task grouping or parameter isolation. (2) Multi-Stage SFT (Random Grouping), where tasks are randomly partitioned into $K = 3$ stages and fine-tuned sequentially, updating all parameters across each stage. (3) Multi-Stage SFT (Heuristic Grouping), where tasks grouped based on perceived similarity (e.g., reasoning tasks grouped together, open-ended tasks grouped together) are fine-tuned sequentially over two stages, with all parameters updated during each stage.

Implement details: All experiments use the LLaMA-2-7B (Touvron et al., 2023), Mistral-8B (Jiang et al., 2024), Gemma-9B (Team et al., 2025), and Qwen2-7B (Yang et al., 2024) as the base language model. Fine-tuning is performed 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. The main SFT stages involve training for three epochs on the datasets for each stage. For DPI, core parameter identification is conducted through probe fine-tuning runs lasting one epoch per task ($E_{\text{probe}} = 1$). DPI hyperparameters are set to a core percentage of $p = 1\%$ and a similarity threshold $\tau = 0.1$. Task groups derived by DPI are randomly ordered for multi-stage training. Masked fine-tuning is applied during each stage, leveraging the dynamic freezing mechanism to preserve task-specific core parameter regions. All experiments are performed on machines equipped with eight NVIDIA A100 GPUs (80GB).

B Related Work

B.1 Supervised Fine-Tuning and Instruction Tuning

Supervised Fine-Tuning (SFT) is a prevalent technique for specializing pre-trained LLMs (Brown et al., 2020; Devlin et al., 2019; Raffel et al., 2020) for desired downstream behaviors. Instruction tuning (Wei et al., 2021; Sanh et al., 2021; Chung et al., 2024; Zheng et al., 2022; Ma et al., 2022; Wu et al., 2025a; Ouyang et al., 2022; Zhang et al., 2024; Anthropic, 2024; Peng et al., 2023; Fei et al., 2022; Liu et al., 2023a,b; Xue et al., 2024) a specific form of SFT, leverages datasets formatted as instructions and responses to enhance model controllability and generalization to unseen tasks. Standard SFT often involves training on a mixture of data from various tasks (Longpre et al., 2023; Chung et al., 2022; Wei et al., 2022; Lu et al., 2023; Wu et al., 2025b,c), typically applying updates across the entire parameter space. While effective for general adaptation, this approach can struggle when task objectives within the SFT data mixture conflict, leading to the performance trade-offs ("seesaw effect") discussed in Section 1. Our work diverges from this standard practice by proposing a method to selectively update parameters based on their identified relevance to specific tasks within the SFT process, thereby directly addressing the negative consequences of indiscriminate parameter updates.

B.2 Task Interference and Knowledge Retention

Task interference has been a persistent challenge in multi-task and sequential learning paradigms. Specifically, training sequentially on multiple tasks often leads to catastrophic forgetting (McCloskey and Cohen, 1989; Wu et al., 2024), where knowledge acquired in earlier stages is overwritten or degraded by subsequent updates. This problem has been studied extensively in the context of smaller neural architectures and motivated approaches such

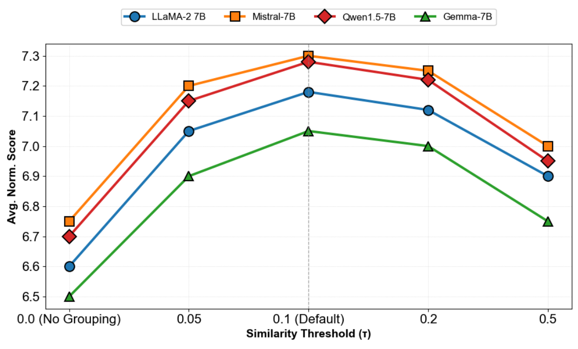

Figure 2: Ablation study on the similarity threshold τ across different base models (with $p = 5\%$). Scores are Avg. Norm. Score.

as regularization (Kirkpatrick et al., 2017; Zhang et al., 2025), replay-based methods (Rolnick et al., 2019; Liu and Wang, 2024), and episodic memory mechanisms (López-Paz and Ranzato, 2017; Guo et al., 2018; Zamir et al., 2018). While effective for small-scale models, extending these techniques to the massive parameter spaces of LLMs is non-trivial. Gradient-based methods have gained popularity for balancing task objectives. GradNorm (Chen et al., 2018; Lopez-Paz and Ranzato, 2022) adjusts task-specific gradient magnitudes, while PCGrad (Yu et al., 2020; Yang et al., 2023) selectively projects conflicting gradients to mitigate interference during multi-task learning. These approaches focus primarily on optimizing task gradients without addressing the root cause of parameter-level contention. Modular solutions, such as adapter fusion (Pfeiffer et al., 2020; Fedus et al., 2022; Xu et al., 2020; Li and Hoiem, 2017; Liang et al., 2019b; Wang et al., 2022; Liang et al., 2019a), assign independent modules to tasks, allowing architectural separation. Despite their advantages, such methods introduce added complexity and may not scale gracefully to hundreds of tasks. Our work departs from these paradigms by taking a parameter-centric approach. Rather than manipulating gradients or enforcing modularity, DPI directly quantifies and isolates the "core parameters" for each task based on update magnitudes.

B.3 Multi-Stage Fine-Tuning and Dynamic Scheduling

Multi-stage fine-tuning frameworks have been widely adopted to tackle task heterogeneity in supervised tuning scenarios. These approaches often heuristically group tasks into stages based on shared characteristics, such as similarity or difficulty, and train sequentially across multiple phases (Ouyang et al., 2022; Wei et al., 2021). While multi-stage frameworks can mitigate direct gradient conflict by separating tasks temporally, they do not account for overlaps in shared parameter usage. As a result, tasks in later stages can destructively overwrite knowledge embedded in parameters critical to earlier tasks, exacerbating catastrophic forgetting. Multi-task fine-tuning strategies emphasize concurrent training on several tasks to enable shared representations (Caruana, 1997). However, multi-task learning encounters challenges in balancing competing task gradients due to loss imbalance or gradient directionality. Techniques combining shared and task-specific spaces, such as (Stickland and Murray, 2020) and (Zhang et al., 2021), attempt to allocate independent regions of the model to different tasks. These approaches maintain task separation but often constrain model capacity and reduce the ability to leverage shared knowledge across tasks effectively. Dynamic task scheduling has emerged as a promising approach to improve multi-task and multi-stage fine-tuning. For instance, (Chen et al., 2020) proposed task prioritization based on difficulty, enabling the model to iteratively refine its understanding across task sequences. Similarly, (Aribandi et al., 2021) presented heuristic strategies for task grouping and ordering to reduce task conflict. Although these methods improve task alignment through better scheduling, they still treat task interactions primarily at a coarse data level and do not address task-specific parameter differentiation within the LLMs. Our approach, Dynamic Parameter Isolation (DPI), extends beyond these advancements by performing explicit parameter-level disentanglement. Unlike static task scheduling, our method dynamically identifies and freezes task-specific parameter regions during multi-stage fine-tuning. Tasks with overlapping parameter regions are grouped into joint training stages to maximize synergy, while disjoint tasks are staged sequentially with frozen core parameters from earlier stages. By aligning task scheduling explicitly with parameter sensitivity, DPI substantially mitigates destructive interference without the need for heuristic task grouping or modular constraints.

B.4 Parameter Heterogeneity and Isolation

The notion of parameter heterogeneity, where different parameters within a model contribute disproportionately to learning specific tasks, has been explored in the contexts of orthogonal weights, sparse updates, and parameter sharing (Neyshabur et al., 2015; Frankle and Carbin, 2019; Mu and Lin, 2025). Inspired by these findings, parameter-efficient fine-tuning methods like adapters (Houlsby et al., 2019; Frantar and Alistarh, 2023) and LoRA (Hu et al., 2021; Solano et al., 2024; Song et al., 2022; Liu et al., 2024; Xue et al., 2023; Li et al., 2024b,a) leverage this heterogeneity by introducing task-dedicated parameter subspaces. Such methods demonstrate that task-specific isolation can effectively reduce interference, but they generally require additional parameters, limiting scalability in resource-constrained applications. A related line of research investigates parameter reuse and specialization within transfer learning and continual learning. (Parmar et al., 2024; Li and Liang, 2021) explored weight specialization during pretraining and task adaptation but did not explicitly quantify task-specific parameter regions. Conceptually closer to our work, (Qi et al., 2024) introduced task-sensitive routing to partition parameter updates among tasks dynamically. While this approach focuses on modular task routing, our framework operates directly on model parameter sensitivity and exploits organic updates of pre-trained LLMs. DPI introduces a principled, data-driven approach to identifying and preserving task-specific core parameter regions within the same model architecture.