GAST: Gradient-aligned Sparse Tuning of Large Language Models with Data-layer Selection

Kai Yao[1], **Zhenghan Song**[2], **Kaixin Wu**[1], **Mingjie Zhong**[1],
Danzhao Cheng[1], **Zhaorui Tan**[3], **Yixin Ji**[4], **Penglei Gao**[5*],
[1]Ant Group [2]Cornell University [3]University of Liverpool
[4]Soochow University [5]Cleveland Clinic Lerner Research Institution
jiumo.yk@antgroup.com, gaop@ccf.org

Abstract

Parameter-Efficient Fine-Tuning (PEFT) has become a key strategy for adapting large language models, with recent advances in sparse tuning reducing overhead by selectively updating key parameters or subsets of data. Existing approaches generally focus on two distinct paradigms: layer-selective methods aiming to fine-tune critical layers to minimize computational load, and data-selective methods aiming to select effective training subsets to boost training. However, current methods typically overlook the fact that different data points contribute varying degrees to distinct model layers, and they often discard potentially valuable information from data perceived as of low quality. To address these limitations, we propose Gradient-aligned Sparse Tuning (GAST), an innovative method that simultaneously performs selective fine-tuning at both data and layer dimensions as integral components of a unified optimization strategy. GAST specifically targets redundancy in information by employing a layer-sparse strategy that adaptively selects the most impactful data points for each layer, providing a more comprehensive and sophisticated solution than approaches restricted to a single dimension. Experiments demonstrate that GAST consistently outperforms baseline methods, establishing a promising direction for future research in PEFT strategies.

1 Introduction

Large Language Models (LLMs) such as GPT-3 (Brown et al., 2020) and LLaMA (Touvron et al., 2023a) form the backbone of modern NLP, yet their enormous size results in significant computational and memory overhead during full fine-tuning on downstream tasks (Howard and Ruder, 2018). To tackle this challenge, Parameter-Efficient Fine-Tuning (PEFT) methods have emerged as a mainstream solution, allowing users to adapt large models with significantly reduced resource overhead by tuning a subset of parameters (Houlsby et al., 2019; Hu et al., 2021; He et al., 2022; Liu et al., 2024). This PEFT strategy is largely guided by human-designed heuristics and fails to account for task-specific domain differences and nuances, which constrains its effectiveness across diverse downstream applications (Yao et al., 2024). Despite these advantages, many PEFT methods still suffer from inefficiencies during training, often requiring substantial computation that undermines their intended performance benefits.

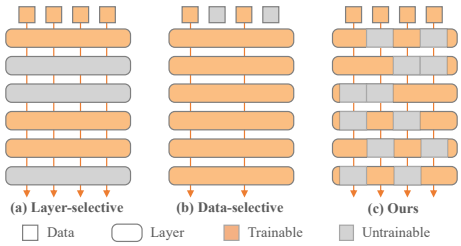


Figure 1: Difference among (a) layer-selective methods, (b) data-selective methods, and (c) our method. layer-selective methods generate a subset of all layers to be updated with all mini-batch data. Data-selective methods utilize partial mini-batch data to train all layers. Our GAST selects a different subset of data for each layer.

To further improve the efficiency of model training without compromising performance under the PEFT framework, numerous studies have been proposed to reduce unnecessary computation and enhance learning dynamics by integrating sparse tuning. One prominent line of research focuses on layer selection, which posits that not all layers in an LLM are equally important for all training updates (Kaplun et al., 2023; Pan et al., 2024; Yao et al., 2024). These approaches seek to reduce redundancy, such as optimizer memory, activation memory, and gradient memory, by estimating the importance score of each layer and selectively activating or updating only a subset of layers during training. While effective, most existing layer-wise

*Corresponding authors.

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 4401–4416

methods apply a uniform layer configuration to all data samples within one mini-batch, implicitly assuming equal importance across samples and neglecting the inherent heterogeneity of data. Consequently, they may underutilize the representational capacity of the model for more complex or atypical samples. Another influential direction centers on data selection, based on the observation that real-world datasets often contain large amounts of low-quality, redundant, or biased information (Wang et al., 2024b,a). By identifying and selecting a subset of informative data points, these methods aim to accelerate training and improve generalization. However, such approaches typically discard low-quality data entirely, potentially overlooking valuable information embedded in those seemingly uninformative examples—information that may become useful in later stages of learning or contribute to model robustness.

Although both layer selection and data selection offer promising routes for effective training and performance improvement, each suffers from critical limitations mentioned above when treated in isolation. One major empirical observation in this paper is that variations in data across tasks and domains might cause the phenomenon that different data points could make distinct and layer-specific contributions to model optimization. We hypothesize this is due to the fact that each layer of an LLM tends to capture different levels of semantic information. Therefore, using the whole dataset for all-layer fine-tuning may lead to gradient conflicts, leading to performance degradation. There remains a significant opportunity to develop more adaptive and fine-grained mechanisms that consider the interaction between data and model structure.

Motivated by the limitations of existing PEFT methods and the aforementioned observation, we investigate the data sparsity among the layers of LLMs and explore the interaction between data complexity and model depth. Building on this insight, we present a theoretical framework that formally demonstrates that both layer selection and data selection are sub-optimal strategies to a more general joint selection paradigm. Thus, we propose a novel method, **G**radient-**a**ligned **S**parse **T**uning (GAST), which simultaneously performs layer-level and data-level selection. Fig. 1 shows the overall difference of our method compared to the layer-selective and data-selective methods. Specifically, GAST could dynamically sample a subset of data points that are most informative to that specific layer's update and compute the gradient of the selected data points as the measurement. This enables us to preserve useful training signals from data points that might be discarded on other layers and ensures that each layer is trained on the most relevant and impactful samples. As a result, GAST achieves both layer-level sparsity and data-level importance, enhancing computational scalability while maintaining or even improving downstream task performance.

In summary, our contributions are as follows:

- We develop a theoretical foundation to demonstrate that both layer-level and data-level selection are sub-optimal to our hybrid data-layer selective sparse tuning.
- We propose a batch-level strategy that dynamically selects both data points and model layers to facilitate sparse training, effectively accelerating convergence and improving model performance.
- Extensive experiments across multiple LLMs and downstream tasks demonstrate that our proposed method achieves consistently better performance and faster convergence compared to existing PEFT approaches.

2 Related Work

2.1 Parameter-efficient Fine-tuning

PEFT has emerged as a dominant approach for adapting LLMs to downstream tasks while mitigating the prohibitive computational and memory costs of full fine-tuning. Early PEFT methods, such as adapter-based models (Houlsby et al., 2019; He et al., 2022; Lei et al., 2024), introduced small trainable bottleneck layers, allowing for task adaptation with minimal parameter updates. More recent methods like LoRA (Hu et al., 2021) and DoRA (Liu et al., 2024) further reduced the total number of trainable parameters. LoRA freezes the original model weights and injects low-rank trainable matrices into each layer's weight update path, while DoRA further improves flexibility by decoupling the low-rank adaptation into separate scaling and shifting operations. Other techniques explore prompt-based tuning by tuning soft prompts while keeping the model frozen (Li and Liang, 2021; Liu et al., 2022). While PEFT methods have demonstrated strong empirical performance across diverse tasks, they typically assume uniform importance across training data and model layers, often failing to exploit the heterogeneity inherent in both.

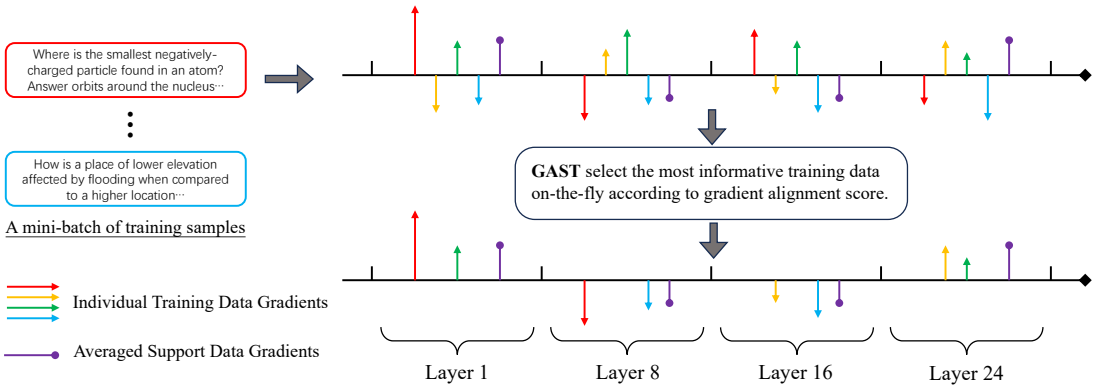


Figure 2: Overall of our proposed Gradient-aligned Sparse Tuning (GAST). During training, every mini-batch exhibits gradient conflicts both among data samples and across model layers. To mitigate these conflicts, GAST uses the gradient of the support set to decide which individual sample should be used to update each layer. This data-layer selection reduces gradient interference and thereby improves both convergence speed and generalization performance.

Our work builds on this foundation by introducing a dynamic mechanism that jointly considers layer-level and data-level sparsity, offering a more flexible strategy.

2.2 Layer-wise Sparse Tuning

Many studies have demonstrated that not all layers in an LLM contribute equally to task-specific adaptation and have proposed selectively fine-tuning a subset of layers to improve training efficiency (Fan et al., 2021; Sajjad et al., 2023; Elhoushi et al., 2024). For instance, Kaplun et al. (2023) employs a greedy search strategy to identify the optimal layers for fine-tuning, but this process incurs substantial computational cost and initialization time. In the work of (Pan et al., 2024), the authors develop a new optimization strategy to accelerate training by randomly selecting a subset of layers. In addition, Yao et al. (2024) utilizes a reinforcement learning strategy to dynamically select the most important subsets with layer-wise importance scoring, achieving better performance and reducing memory as well. However, they typically treat all data points uniformly and apply the same subsets to every input regardless of its complexity or relevance, which limits their ability to exploit the diverse learning signals present across the training corpus.

2.3 Data-wise Sparse Tuning

Recently, increasing attention has been directed toward developing methods for selecting data before training foundation models. Xia et al. (2024) introduces an optimizer-aware algorithm that estimates data influence under the Adam optimizer and performs low-rank gradient similarity search to select instruction data. Other methods investigate online batch selection to enhance the training of LLMs based on gradient norm or maximum sample loss of "hard samples" (Katharopoulos and Fleuret, 2018; Jiang et al., 2019), or explore using additional reference models to more accurately estimate the importance of samples (Deng et al., 2023). Furthermore, Wang et al. (2024b) applies a greedy algorithm to optimize the data batch quality approximated by Taylor expansion, efficiently capturing true data informativeness. However, they still treat layer selection uniformly across inputs, which may cause gradient conflicts or insufficient information capturing for specific layers.

3 Method

In this section, we first present our theoretical motivation and show that the alignment of training gradients with a support gradient can guide more effective parameter updates. Next, we introduce our method, Gradient-aligned Sparse Tuning (GAST). GAST dynamically selects adapter parameters that are most positively aligned with the support gradient, improving performance and generalization.

3.1 Theoretical Motivation

We consider a PEFT-equipped LLM as $\mathcal{M}_{\Theta+\Delta}$, where Θ are the frozen backbone weights and Δ are the trainable adapter parameters. Given a small support set $\mathcal{D}_{\text{sup}}$, the gradient induced by the support set at step t for layer i is $g^{(i)}_{t,\text{sup}} = \mathbb{E}_{\mathcal{D}_{\text{sup}}} \nabla \ell(\Theta, \Delta_t; \mathcal{D}_{\text{sup}})$. The gradient induced by

a training example x_j is $g_{t,j}^{(i)} = \nabla\ell(\Theta, \Delta_t; x_j)$. Consider the gradient alignment between training gradients and support gradients:

$$\mathrm{sim}(g_{t,j}^{(i)}, g_{t,\text{sup}}^{(i)}) = \frac{\langle g_{t,j}^{(i)}, g_{t,\text{sup}}^{(i)} \rangle}{||g_{t,j}^{(i)}||\,||g_{t,\text{sup}}^{(i)}||}.$$

A positive alignment implies that updating parameters using sample j will effectively reduce support-set loss, while a negative alignment indicates gradient conflict as shown in Fig. 2. For a layer-wise spare tuning strategy, it uses all training data on the selected layers. We define its gradient aggregation for layer i as: $g_{t,\text{layer}}^{(i)} = \frac{1}{|\mathcal{D}_{\text{train}}|}\sum_{j\in\mathcal{D}_{\text{train}}} g_{t,j}^{(i)}$. Similarly, for the data-wise selection strategy, we have the following formula: $g_{t,\text{data}}^{(i)} = \frac{1}{|\mathcal{D}_{\text{sub}}|}\sum_{j\in\mathcal{D}_{\text{sub}}} g_{t,j}^{(i)}$, where $\mathcal{D}_{\text{sub}}$ is the selected subset of the training data for each layer. To leverage the superiority of our hybrid data-layer selective sparse tuning, we define the dynamically selected subset at layer i as: $\mathcal{D}_+^{(i)} = \{x_j : \langle g_{t,j}^{(i)}, g_{t,\text{sup}}^{(i)} \rangle > 0\}$. Thus the aggregated gradient at layer i is: $g_{t,\text{hybrid}}^{(i)} = \frac{1}{|\mathcal{D}_+^{(i)}|}\sum_{j\in\mathcal{D}_+^{(i)}} g_{t,j}^{(i)}$.

Consider the magnitude of gradient projection onto the support gradient $\langle g_{t,\cdot}^{(i)}, g_{t,\text{sup}}^{(i)} \rangle$, there exists one example $x_{j'} \in \mathcal{D}_{\text{sub}}, \quad \text{s.t.} \quad \langle g_{t,j'}^{(i)}, g_{t,\text{sup}}^{(i)} \rangle < 0$ for the data-selective strategy, which selected fixed subset for the training data. For the layer-selective strategy using all training data, there always exists a gradient conflict having a negative gradient alignment. However, the hybrid selection achieves strictly greater effective gradient magnitude toward support-set minimization with each term $\langle g_{t,j}^{(i)}, g_{t,\text{sup}}^{(i)} \rangle > 0$ for $x_j \in \mathcal{D}_+^{(i)}$. With the assumptions that 1) a non-empty set of positively aligned per-sample gradients; 2) negative-alignment gradients do not dominate in magnitude, and 3) the baseline data subset $\mathcal{D}_{\text{sub}}$ is fixed and not constructed using gradient alignment. Thus, we could have:

$$\langle g_{t,\text{hybrid}}^{(i)}, g_{t,\text{sup}}^{(i)} \rangle \geq \max\{\langle g_{t,\text{layer}}^{(i)}, g_{t,\text{sup}}^{(i)} \rangle, \langle g_{t,\text{data}}^{(i)}, g_{t,\text{sup}}^{(i)} \rangle\}. \tag{1}$$

Consider a smoothness assumption on the loss function (detailed proof in the supplementary):

Lemma 1 (L-Smoothness) *Let $\ell(\Delta)$ be an L-smooth objective with respect to Δ. At iteration t, let Δ_t denote the current parameters and let g_t be a stochastic gradient estimator satisfying*

$$\mathbb{E}[g_t \mid \Delta_t] = \nabla\ell(\Delta_t). \tag{2}$$

With step size $\eta_t > 0$, the conditional expectation of the loss satisfies

$$\mathbb{E}[\ell(\Delta_{t+1}) \mid \Delta_t] \leq \ell(\Delta_t) - \eta_t\,\mathbb{E}\big[\langle \nabla\ell(\Delta_t),\, g_t\rangle \,\big|\, \Delta_t\big] + \frac{L\eta_t^2}{2}\,\mathbb{E}\big[\|g_t\|^2 \,\big|\, \Delta_t\big]. \tag{3}$$

Here $C > 0$ is the Lipschitz constant. In particular, for fixed η_t and bounded $\mathbb{E}[\|g_t\|^2 \mid \Delta_t]$, any strategy that yields a larger $\mathbb{E}[\langle \nabla\ell(\Delta_t),\, g_t\rangle \mid \Delta_t]$ (i.e., better alignment with the true gradient) leads to a larger expected one-step decrease in the loss.

Since our method maximizes the projection $\langle \nabla\ell(\Delta_t), g\rangle$, we obtain the fastest decrease in loss per step given the same support set and learning rate. For each layer i, the expected reduction in loss is largest for our hybrid strategy based on Eq. 1, due to greater gradient alignment and magnitude. Formally, the per-step expected loss reduction satisfies:

$$\ell(\Delta_{t+1,\text{hybrid}}^{(i)}) \leq \min\{\ell(\Delta_{t+1,\text{layer}}^{(i)}), \ell(\Delta_{t+1,\text{data}}^{(i)})\}. \tag{4}$$

3.2 Gradient-aligned Sparse Tuning

Derived from Eq. 4, under such *gradient heterogeneity*, both traditional *layer-selective* and *data-selective* strategies can only achieve sub–optimal solutions when the batch size is larger than one. To overcome this limitation, as shown in Fig. 2, we introduce GAST, which dynamically assigns different adapter layers to different training examples within the same mini-batch, based on their instantaneous gradient information. Concretely, for every mini-batch, we compare the gradients of the training samples with the gradients obtained on a held-out support set, and select the layers whose gradients are the most similar—the whole procedure is performed on-the-fly.

Starting from an initial adapter Δ_0, standard full-adapter training proceeds as

$$\Delta_{t+1} = \Delta_t - \eta_t \sum_{x\in\mathcal{B}_t} \nabla\ell(\Theta, \Delta_t; x), \tag{5}$$

where $\mathcal{B}_t$ is the mini-batch selected at step t and η_t is the learning rate. Instead of updating all adapter parameters for all samples, GAST seeks, for every example $x_j \in \mathcal{B}_t$, a sample-specific subset of adapter parameters $\widehat{\Delta}_{t,j} \in \Delta_t$, such that only the weights in $\widehat{\Delta}_{t,j}$ are updated with gradient from x_j.

Let $\mathcal{X}_{\text{train}}$ and $\mathcal{X}_{\text{sup}}$ denote the training and support sets, respectively. Considering the total differential theory related to the first-order Taylor expansion (Xia et al., 2024) (detailed in the Supplementary):

Algorithm 1 Gradient-aligned Sparse Tuning (GAST)

Require: LLM $\mathcal{M}$ with L layers equipped with PEFT parameterized by Δ, training dataset $\mathcal{D}_{\text{train}}$, support dataset $\mathcal{D}_{\text{sup}}$

Initialize PEFT parameters Δ_0 for all layers.
for all $t = 0, \ldots, T-1$ **do**
  Sampled a random mini-batch $\mathcal{B}_t \sim \mathcal{D}_{\text{train}}$
  for all $i = 0, \ldots, L-1$ **do**
    for all $j = 0, \ldots, |\mathcal{B}_t| - 1$ **do**
      Calculate gradient alignment score $s_{t,j}^{(i)} = \left\langle g_{t,\text{sup}}^{(i)}, g_{t,j}^{(i)} \right\rangle$
    end for
    Calculate the sampling probability $p_{t,j}^{(i)}$ (see Eq. 9)
    Sampling the updating indices $j^*(i)$ (see Eq. 10)
    Update PEFT parameters $\Delta_{t+1}^{(i)} = \Delta_t^{(i)} - \eta_t \nabla g_{t,(j^*(i))}^{(i)}$
  end for
end for

Theorem 1 (Total Differential.) *The sum of the products of each partial derivative and the corresponding small change in the weight variable can estimate the increment in the loss function at a given point.*

Let $w^{(i)}$ be the adapter weights of layer $i \in \{1, \ldots, L\}$, and let $\delta^{(i)}$ be the small change applied to $w^{(i)}$. Then the increment of the support-set loss obeys

$$\mathcal{L}_{t+1} = \mathcal{L}_t + \sum_{i \in \{1,\ldots,L\}} \left\langle \frac{\partial \mathcal{L}_t}{\partial w_i}, \delta_i \right\rangle, \tag{6}$$

where $\mathcal{L}_t = \sum_{x_k \in \mathcal{X}_{\text{sup}}} \ell(\Theta, \Delta_t; x_k)$ denote the support-set loss, $\langle \cdot, \cdot \rangle$ denotes inner product.

For a mini-batch $\mathcal{B}_t = \{x_{t,1}, \ldots, x_{t,|\mathcal{B}_t|}\}$, for every layer i and every sample $x_{t,j} \in \mathcal{B}_t$, we have the layer-wise, sample-specific gradient:

$$g_{t,j}^{(i)} = \nabla_{\Delta^{(i)}} \ell(\Theta, \Delta_t; x_{t,j}). \tag{7}$$

Hence, setting $\delta^{(i)} = -\eta_t g_{t,j}^{(i)}$ and $g_{\text{sup}}^{(i)} = \nabla_{w^{(i)}} \mathcal{L}_t = \partial \mathcal{L}_t / \partial w_i$ we obtain

$$\mathcal{L}_{t+1} - \mathcal{L}_t = -\eta_t \sum_{i=1}^{L} s_{t,j}^{(i)}, \quad s_{t,j}^{(i)} = \left\langle g_{t,\text{sup}}^{(i)}, g_{t,j}^{(i)} \right\rangle, \tag{8}$$

where $s_{t,j}^{(i)}$ denote the gradient alignment score. Note that computing $g_{t,\text{sup}}$ over the support set at every step should be prohibitively expensive. Therefore, we sample a small subset of support data at each iteration, with a batch size much smaller than that of the training batch.

Eq. 6 suggests that, for every layer i, we should pick the training example whose gradient aligns most positively with the current support-set gradient in order to maximize the expected decrease in support-set loss.

To mitigate the risk of overfitting to the support set, we employ a stochastic selection scheme. Specifically, we first compute the sampling probability for each sample in the batch based on the normalized alignment score:

$$p_{t,j}^{(i)} = \frac{\exp((\widehat{s}_{t,j}^{(i)}))}{\sum_k \exp((\widehat{s}_{t,k}^{(i)}))}, \quad \widehat{s}_{t,\cdot}^{(i)} = \text{Norm}(s_{t,\cdot}^{(i)}) \tag{9}$$

where $\text{Norm}(\cdot)$ denotes mean-std normalization over mini-batch.

Next, for each layer i, we sample K indices according to $p_{t,j}^{(i)}$, and denote the index used to update layer i as

$$j^*(i) \sim \text{Categorical}\big(p_{t,1}^{(i)}, \ldots, p_{t,|\mathcal{B}_t|}^{(i)}\big). \tag{10}$$

To achieve data-layer selective updating, only the gradient of the selected data points $x_{t,j^*(i)}$ are used to update layer i, i.e.,

$$\Delta_{t+1}^{(i)} = \Delta_t^{(i)} - \eta_t g_{t,(j^*(i))}^{(i)}. \tag{11}$$

Intuitively, GAST enables each adapter layer to learn only from the most relevant training examples in a mini-batch, thereby making parameter updates more targeted and effective. This data-layer-selective sparse tuning both leverages gradient heterogeneity and promotes better generalization. We detail the complete GAST training procedure in Algorithm 1 for better clarification.

4 Experiments

4.1 Experimental Setup

Models. We selected LLaMA-7B, LLaMA-13B (Touvron et al., 2023b), GPT-J-6B (Wang and Komatsuzaki, 2021), and LLaMA3-8B as the foundational models for downstream task finetuning, considering their widespread use in the research community. For reference, we also report results from ChatGPT (gpt-3.5-turbo) using zero-shot Chain-of-Thought prompting (Wei et al., 2022).

Datasets. We evaluated our method on commonsense and arithmetic reasoning tasks. For commonsense reasoning, we utilized eight subtasks, each with standard training and testing splits:

BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC (Clark et al., 2018), and OBQA (Mihaylov et al., 2018). Following Hu et al. (2023), we aggregated the training sets of all these tasks for joint training and evaluated the fine-tuned models on the official test split for each sub-task. For arithmetic reasoning, we fine-tuned models on the Math10K dataset, which contains math reasoning samples curated by Hu et al. (2023). Evaluation was performed on the official test sets of datasets, including GSM8K (Cobbe et al., 2021), AQuA (Ling et al., 2017), MAWPS (Koncel-Kedziorski et al., 2016), and SVAMP (Patel et al., 2021).

Implementation Details. Our experiments are conducted using the LLM-Adapter framework (Hu et al., 2023). Consistent with prior work, we set the batch size to 16 and the number of training epochs to 3. In GAST, we treat the whole training set as the support set; at every iteration we randomly pick four samples from it to calculate the averaged support-set gradient, and set $K = 8$. For PEFT, we adopt Series Adapter (Houlsby et al., 2019), Parallel Adapter (He et al., 2022), and Low-rank Adapter (LoRA) (Hu et al., 2021). For LoRA, we used a rank of 32 with an alpha of 64, a dropout rate of 0.05, and applied it to the Q, K, V, Up, Down modules. For the Series and Parallel adapters, we employed a bottleneck size of 256, equipping them with Up, Down and Up, Gate modules, respectively. A uniform learning rate of 1e-4 with 100 warmup steps was set for all methods, and a maximum context length of 256 was selected. For the 7B models, training was conducted for 11.5 hours. All experiments were conducted on a workstation equipped with an NVIDIA A100 80GB GPU.

4.2 Baselines

We compare GAST with the following advanced representative adaptive methods:

LISA (Pan et al., 2024): A PEFT approach that sparsely tunes only a single transformer layer, significantly reducing trainable parameters.

AdaLoRA (Zhang et al., 2023): A rank-adaptive PEFT method that can be broadly applied to various reparameterization-based strategies, which dynamically adjusts the rank size of each LoRA.

RST (Yao et al., 2024): This method randomly selects and fine-tunes a subset of transformer layers, thereby reducing trainable parameters.

Method	Adaptiveness	Average
LoRA	-	74.7 (+0.0)
LISA	Layer	75.3 (+0.6)
AdaLoRA	Rank	76.2 (+1.5)
LoRA + RST	Layer	75.8 (+1.1)
LoRA + IST	Layer	76.5 (+1.8)
LoRA + GREATS	Data	76.3 (+1.6)
LoRA + GAST	Data & Layer	**77.5 (+2.8)**

Table 1: Comparison with different adaptive methods.

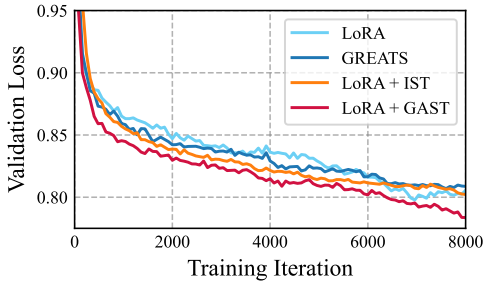


Figure 3: Comparison of model convergence with loss curve.

IST (Yao et al., 2024): The latest layer-selective sparse tuning method, which employs reinforcement learning to rank layer importance.

GREATS (Wang et al., 2024b): A recent online data selection approach that utilizes the calculation of Data Shapley (Wang et al., 2024a) values in an online manner. At each step, GREATS selects the optimal half of the training data greedily.

4.3 Comparison with Adaptive Methods

The comparison results shown in Tab. 1 demonstrate the consistent improvement and effectiveness of our GAST method with other adaptive methods on the commonsense task evaluated on the LLaMA 7B model. The standard LoRA baseline achieves an average score of 74.7, while methods utilizing adaptiveness at either the layer or rank level, such as LISA, RST, and AdaLoRA, achieve promising improvements, with AdaLoRA reaching 76.2 by dynamically adjusting module ranks. Data-selective methods like GREATS also provide notable gains (76.3), and layer-selective methods such as IST further enhance prediction performance (76.5). Among all compared approaches, GAST achieves the highest average score (77.5), indicating that jointly leveraging gradient heterogeneity across both data and layers results in more effective parameter tuning.

To further compare the methods, we visualized the validation loss of different approaches to compare the convergence of the models. As shown

Model	PEFT	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg.
ChatGPT	-	73.1	85.4	68.5	78.5	66.1	89.8	79.9	74.8	77.0
$LLaMA_{7B}$	Series	63.0	79.2	76.3	67.9	75.7	74.5	57.1	72.4	70.8
	Series + IST	66.2	78.3	74.9	72.2	75.9	75.8	59.0	72.2	71.8
	Series + GAST	68.0	79.1	77.8	79.8	78.5	77.4	62.0	74.8	**74.7**
	Parallel	67.9	76.4	78.8	69.8	78.9	73.7	57.3	75.2	72.2
	Parallel + IST	68.4	79.1	77.9	70.0	78.9	81.2	62.3	77.6	74.4
	Parallel + GAST	67.9	81.3	78.4	80.5	79.8	78.2	63.0	77.4	**75.8**
	LoRA	68.9	80.7	77.4	78.1	78.8	77.8	61.3	74.8	74.7
	LoRA + IST	68.7	81.7	77.3	82.7	78.7	80.6	62.4	80.0	76.5
	LoRA + GAST	68.2	81.6	79.4	83.6	82.2	80.4	64.7	79.8	**77.5**
$LLaMA_{13B}$	Series	71.8	83.0	79.2	88.1	82.4	82.5	67.3	81.8	79.5
	Series + IST	72.9	82.2	81.4	87.9	84.0	82.7	69.1	81.1	80.2
	Series + GAST	72.9	82.0	82.8	89.5	85.8	84.2	70.6	82.0	**81.2**
	Parallel	72.5	84.9	79.8	92.1	84.7	84.2	71.2	82.4	81.4
	Parallel + IST	72.6	86.0	79.2	89.1	83.5	84.8	70.6	82.8	81.1
	Parallel + GAST	72.8	86.1	80.6	91.0	85.8	86.0	72.1	84.0	**82.3**
	LoRA	72.1	83.5	80.5	90.5	83.7	82.8	68.3	82.4	80.5
	LoRA + IST	71.5	85.0	81.2	89.1	84.2	84.0	70.1	81.8	80.9
	LoRA + GAST	73.2	84.5	81.9	90.8	85.5	84.6	71.5	82.9	**81.8**
$GPT\text{-}J_{6B}$	LoRA	62.4	68.6	49.5	43.1	57.3	43.4	31.0	46.6	50.2
	LoRA + IST	63.0	63.2	62.9	35.8	39.1	56.8	39.1	51.2	51.4
	LoRA + GAST	63.1	74.4	65.0	49.7	59.9	59.7	45.3	60.2	**59.7**
$LLaMA3_{8B}$	LoRA	70.8	85.2	79.9	91.7	84.3	84.2	71.2	79.0	80.8
	LoRA + IST	72.7	88.3	80.5	94.7	84.4	89.8	79.9	86.6	84.6
	LoRA + GAST	73.6	87.4	81.0	95.2	86.5	90.0	79.8	85.0	**84.8**

Table 2: Comparison across multiple LLMs using different PEFT approaches on eight commonsense reasoning benchmarks. Baseline results for GPT-J and LLaMA are taken from Hu et al. (2023).

Method	GSM8K	AQuA	MAWPS	SVAMP	Avg.
ChatGPT	56.4	38.9	87.4	69.9	63.2
LoRA	61.0	26.4	91.6	74.4	63.4
LoRA + IST	62.8	31.5	89.9	76.3	64.7
LoRA + GAST	66.4	32.7	91.6	79.4	**67.5**

Table 3: Comparison with LLaMA3-8B on four math reasoning datasets.

Setting	Data-layer Selection	Average
LoRA	-	74.7
LoRA + GAST	Random Selection	76.4
LoRA + GAST	Top-k Selection	66.4
LoRA + GAST	Sampling-based Selection	**77.5**

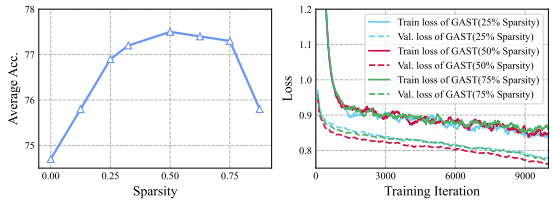
Table 4: The effect of data-layer selection strategy.

in Fig. 3, the data-selective GREATS and layer-selective IST methods significantly outperform the baseline LoRA in the early stages of training but experience fluctuations in the middle stages. This may be due to gradient conflict on data for each layer, which manifests in the middle stages of the model. In contrast, our method consistently surpasses all other methods, demonstrating its effectiveness in overcoming gradient conflict.

4.4 Evaluation of Versatility

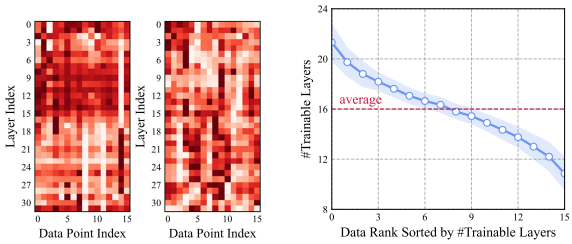
To further demonstrate the broad improvements provided by our method for various PEFT approaches, we conducted experiments on different models, datasets, and PEFT methods. The quantitative results in Tab. 2 present an extensive evaluation of the effectiveness of GAST across a wide range of PEFT settings and LLM backbones. We observe that integrating GAST consistently enhances the performance of various models on commonsense reasoning benchmarks. Taking LLaMA-7B as an example, GAST leads to clear accuracy improvements under all three PEFT strategies (Series, Parallel, and LoRA). For instance, on the challenging HellaSwag dataset, LoRA+GAST improves accuracy from 78.1% to 83.6% compared to standard LoRA, representing a notable advance. This trend holds on other competitive datasets such as WinoGrande and ARC-c as well. Additionally, GAST brings significant gains to GPT-J-6B, where the average accuracy increases by over 9 points over the LoRA baseline. These improvements are consistent across almost all datasets and model configurations, confirming the robustness and broad applicability.

We further assess GAST's effectiveness on a se-



(a) Data-layer Sparsity (b) Loss curve

Figure 4: Impact of data-layer sparsity in GAST.



(a) Sampling probability (b) Num. of sampled layers

Figure 5: (a) Visualization of the sampling probability of mini-batch data points across layers on two different iterations. A deeper red color indicates a higher probability. (b) Distribution of the number of layers each data point is trained on within a mini-batch.

ries of math reasoning benchmarks, as summarized in Tab. 3. GAST demonstrates consistent and notable improvements when compared to IST. For example, when applying GAST on LLaMA3-8B with LoRA, we see increases not only in the average score (from 63.4% with LoRA to 67.5% with LoRA+GAST), but also robust improvements on individual datasets such as GSM8K and SVAMP. These results indicate that GAST is not only beneficial for commonsense reasoning, but also generalizes well to tasks requiring complex mathematical reasoning. The observed gains across such diverse benchmarks further highlight the versatility and practical value of GAST in enriching parameter-efficient fine-tuning of large language models.

4.5 Analytical Studies

Ablation Study. We conducted experiments to evaluate the effects of gradient-aligned sparse tuning by training the LLaMA 7B model with LoRA on a commonsense task and reporting the average accuracy. As shown in Tab. 4, using random data-layer selection led to significant performance improvements, consistent with the findings of IST (Yao et al., 2024) that training with fewer parameters can enhance generalization. However, adopting a top-k selection strategy did not improve performance. This may be due to the small mini-batch drawn from the support set, which fails to capture the overall gradient distribution of the test set. Finally, we employed a sampling-based selection approach, which yielded the best results and further demonstrated the effectiveness of GAST.

Impact of Sparsity in GAST. We conducted experiments to evaluate the effect of data-layer sparsity in GAST by training the LLaMA 7B model with LoRA on a commonsense reasoning task. As shown in Fig. 4(a), we gradually increased the data-layer sparsity from 0.0 to 0.875, reporting the averaged test accuracies. Here, a sparsity of 0.0 corresponds to the standard LoRA setting, while 0.5 is our default configuration. The experimental results indicate that when the sparsity is set to an extremely high value (0.875), the network becomes overly sparse and performance drops. Nevertheless, even in this highly sparse regime, the model still outperforms the baseline standard LoRA, which suggests the effectiveness of data-layer selective utilization. Between sparsity values of 0.25 and 0.75, the performance remains relatively stable, reaching its peak at 0.5, after which it gradually declines. Fig. 4(b) shows the training and validation loss curves corresponding to different sparsity levels. It can be observed that the denser setting (i.e., with 75% sparsity) exhibits a faster decrease in training loss during the initial stage. However, both excessively high and low sparsity levels fail to yield a lower validation loss. This may be because low sparsity can cause gradient conflicts, while high sparsity results in insufficient information being retained. These results suggest that a sparsity level of 0.5 strikes a balance between preserving important information and avoiding gradient conflicts.

Distribution of Sampled Layers. To gain a deeper understanding of our method, we visualize the sampling probabilities of 16 data points within a mini-batch, across the 32 layers of LLaMA-7B. A higher sampling probability indicates a higher gradient alignment score. As shown in Fig 5(a), different data points exhibit distinct probability distributions across the layers. For instance, in the left figure, the 8th and 11th data points have higher probabilities in the shallow layers, whereas the 14th data point shows higher probabilities in the deeper layers. This demonstrates that our approach can effectively estimate the contribution of each data point to different layers. Next, we visualize the number of sampled layers for each data point per iteration. As shown in Fig 5(b), although the spar-

sity is set to 50%, the most important data points are trained in up to 70% of the layers, while the least important ones are only trained in 30% of the layers. This indicates that our GAST dynamically allocates the number of layers fine-tuned for each data point according to its gradient alignment.

5 Conclusion

In this work, we introduced GAST, a novel PEFT framework that jointly considers both data point and layer selection to finetune LLMs. The proposed GAST dynamically assigns informative data to specific layers based on gradient alignment with a holdout support set. This fine-grained, data-layer selection mechanism effectively mitigates gradient conflicts, improving both convergence and performance. We theoretically demonstrated that our hybrid selection strategy yields strictly better gradient alignment and faster convergence compared to existing layer-wise or data-wise sparse tuning methods. We believe this work opens promising avenues for more adaptive and generalizable tuning paradigms for large models.

Limitations

There are two limitations in this work. First, similar to GREATS, our method cannot simultaneously reduce both memory usage and computational cost due to engineering optimization constraints, even though it could potentially achieve higher performance. Second, because of limited resources, we were unable to validate larger language models such as LLaMA 3 70B. Whether larger models require sparser fine-tuning remains unknown, and we leave this as future work.

References

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

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

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Zhijie Deng, Peng Cui, and Jun Zhu. 2023. Towards accelerated model training via bayesian data selection. In *Advances in Neural Information Processing Systems*, volume 36, pages 8513–8527.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, et al. 2024. Layer skip: Enabling early exit inference and self-speculative decoding. *arXiv preprint arXiv:2404.16710*.

Chun Fan, Jiwei Li, Xiang Ao, Fei Wu, Yuxian Meng, and Xiaofei Sun. 2021. Layer-wise model pruning based on mutual information. *arXiv preprint arXiv:2108.12594*.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

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.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Zhiqiang Hu, Yihuai Lan, Lei Wang, Wanyu Xu, Ee-Peng Lim, Roy Ka-Wei Lee, Lidong Bing, and Soujanya Poria. 2023. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Empirical Methods in Natural Language Processing*.

Angela H Jiang, Daniel L-K Wong, Giulio Zhou, David G Andersen, Jeffrey Dean, Gregory R Ganger, Gauri Joshi, Michael Kaminksy, Michael Kozuch, Zachary C Lipton, et al. 2019. Accelerating deep learning by focusing on the biggest losers. *arXiv preprint arXiv:1910.00762*.

Gal Kaplun, Andrey Gurevich, Tal Swisa, Mazor David, Shai Shalev-Shwartz, and Eran Malach. 2023. Less is more: Selective layer finetuning with subtuning. *arXiv preprint arXiv:2302.06354*.

Angelos Katharopoulos and François Fleuret. 2018. Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, pages 2525–2534. PMLR.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of NAACL*, pages 1152–1157.

Tao Lei, Junwen Bai, Siddhartha Brahma, Joshua Ainslie, Kenton Lee, Yanqi Zhou, Nan Du, Vincent Zhao, Yuexin Wu, Bo Li, et al. 2024. Conditional adapters: Parameter-efficient transfer learning with fast inference. *Advances in Neural Information Processing Systems*, 36.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: Weight-decomposed low-rank adaptation. In *International Conference on Machine Learning*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association of Computational Linguistics*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Empirical Methods in Natural Language Processing*.

Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. 2024. Lisa: Layerwise importance sampling for memory-efficient large language model fine-tuning. *arXiv preprint arXiv:2403.17919*.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of NAACL*, pages 2080–2094.

Hassan Sajjad, Fahim Dalvi, Nadir Durrani, and Preslav Nakov. 2023. On the effect of dropping layers of pre-trained transformer models. *Computer Speech & Language*, 77:101429.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*.

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

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

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Jiachen T Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. 2024a. Data shapley in one training run. *arXiv preprint arXiv:2406.11011*.

Jiachen T Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. 2024b. Greats: online selection of high-quality data for llm training in every iteration. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, pages 131197–131223.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*.

Kai Yao, Penglei Gao, Lichun Li, Yuan Zhao, Xiaofeng Wang, Wei Wang, and Jianke Zhu. 2024. Layer-wise importance matters: Less memory for better performance in parameter-efficient fine-tuning of large language models. In *Findings of the Association*

for Computational Linguistics: EMNLP 2024, pages 1977–1992.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*. Openreview.

A More Details of Our Method

A.1 Proof of L-Smoothness

Consider the loss function $\ell(\cdot)$, we have the following assumption:

Assumption 1 ***(Lipschitz-continuous objective gradients)***
Let ℓ be a continuously differentiable loss function and the gradient function of ℓ as $\nabla\ell$ is Lipschitz continuous with Lipschitz constant $L > 0$, i.e.,

$$||\nabla\ell(\Delta_{t+1}) - \nabla\ell(\Delta_t)||_2 \leq L||\Delta_{t+1} - \Delta_t||_2 \quad for\ all\ \{\Delta_{t+1}, \Delta_t\} \subset \mathbb{R}^d. \tag{12}$$

This assumption ensures that the gradient of ℓ does not change arbitrarily quickly with respect to the parameter vector. Then we could have the following formular:

$$\begin{aligned}\ell(\Delta_{t+1}) \leq & \ell(\Delta_t) + \nabla\ell(\Delta_t)^T(\Delta_{t+1} - \Delta_t) + \\ & \frac{1}{2}L||\Delta_{t+1} - \Delta_t||_2^2 \\ & for\ all \quad \{\Delta_{t+1}, \Delta_t\} \subset \mathbb{R}^d.\end{aligned} \tag{13}$$

Under Assumption 1, we can obtain:

$$\begin{aligned}&\ell(\Delta_{t+1}) = \ell(\Delta_t) + \int_0^1 \frac{\partial\ell(\Delta_t + s(\Delta_{t+1} - \Delta_t))}{\partial s}\, ds \\ &= \ell(\Delta_t) + \int_0^1 \nabla\ell(\Delta_t + s(\Delta_{t+1} - \Delta_t))^T(\Delta_{t+1} - \Delta_t)\, ds \\ &= \ell(\Delta_t) + \nabla\ell(\Delta_t)^T(\Delta_{t+1} - \Delta_t)) + \\ &\int_0^1 [\nabla\ell(\Delta_t + s(\Delta_{t+1} - \Delta_t)) - \nabla\ell(\Delta_t)]^T(\Delta_{t+1} - \Delta_t)\, ds \\ &\leq \ell(\Delta_t) + \nabla\ell(\Delta_t)^T(\Delta_{t+1} - \Delta_t) + \\ &\int_0^1 L||s(\Delta_{t+1} - \Delta_t)||_2||\Delta_{t+1} - \Delta_t||_2\, ds,\end{aligned} \tag{14}$$

which results in the inequality 13. Then, according to the iterates of Stochastic Gradient, we can have:

$$\begin{aligned}\ell(\Delta_{t+1}) - \ell(\Delta_t) &\leq \nabla\ell(\Delta_t)^T(\Delta_{t+1} - \Delta_t) \\ &\quad + \frac{1}{2}L||\Delta_{t+1} - \Delta_t||_2^2 \\ &\leq -\eta_t\nabla\ell(\Delta_t)^T g(\Delta_t, x_t) \\ &\quad + \frac{1}{2}\eta_t^2 L||g(\Delta_t, x_t)||_2^2.\end{aligned} \tag{15}$$

This inequality 15 shows that the expected decrease in the objective function yielded by the t-th step is bounded above by two components. One is the expected directional derivative of ℓ at Δ_t along $-g(\Delta_t, x_t)$. The other is the second moment of $g(\Delta_t, x_t)$.

A.2 Proof of Total Differential

Given a function $f(x_1, x_2, \ldots, x_n)$ with variables $x_1, x_2, \ldots, x_n$, its total differential df can be expressed as:

$$df = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} dx_i, |dx_i| < \epsilon, \tag{16}$$

where ϵ is a small value. By considering the sample as a constant and treating w as the input x, with $f = \ell(w)$ representing the loss calculation, we can derive the total differential of $\ell(w)$ as follows:

$$\partial\ell(w) = \sum_{i=n_1}^{n_2} \left\langle \frac{\partial\ell}{\partial w_i}, dw_i \right\rangle. \tag{17}$$

dw_i can be considered as the δ_i between w_i and $\hat{w}_i$. Therefore, we have:

$$\ell(\hat{w}) - \ell(w) = \sum_{i=n_1}^{n_2} \left\langle \frac{\partial\ell}{\partial w_i}, \delta_i \right\rangle, \tag{18}$$

$\langle\cdot,\cdot\rangle$ denotes inner product.

A.3 Efficient Implement of GAST

In our method, one potential challenge lies in implementing data-layer selective gradient updating. We propose two approaches for implementing our algorithm: one that is *computation-efficient* and the other that is *memory-efficient*.

For the computation-efficient implementation, our goal is to minimize the amount of computation as much as possible. Specifically, for each Linear in every transformer layer during the backward pass, we save the intermediate per-sample's gradients. After sampling the training data points for that layer, we select and sum the saved mini-batch per-sample gradients. This approach introduces almost no additional computation, allowing the algorithm to be completed in a single forward-backward pass, thus maintaining computational efficiency. However, since it requires saving the per-sample gradients for all Linears in at least one transformer layer, it comes at the cost of increased memory consumption.

For the memory-efficient implementation, given that the computed gradient alignment score is a vector and only the sum of the scores for each Linear in every transformer layer is needed, we can reduce memory demands by releasing memory immediately after computing the score. An additional forward-backward pass can then be used to

Dataset	# Train	# Test	Answer
Commonsense	170K	-	-
BoolQ	9.4K	3,270	Yes/No
PIQA	16.1K	1,830	Option
SIQA	33.4K	1,954	Option
HellaSwag	39.9K	10,042	Option
WinoGrande	63.2K	1,267	Option
ARC-e	1.1K	2,376	Option
ARC-c	2.3K	1,172	Option
OBQA	5.0K	500	Option
Math10K	10K	-	-
GSM8K	8.8K	1,319	Number
AQuA	100K	254	Option
MAWPS	-	238	Number
SVAMP	-	1,000	Number

Table 5: The statistics of datasets for evaluation. # Train and # Test denote the number of training and test samples respectively.

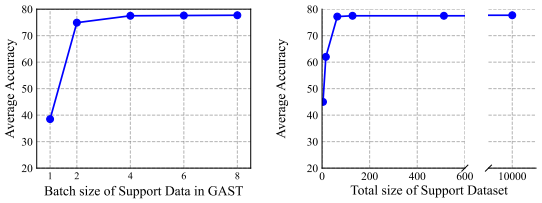


Figure 6: Additional experiment on support set.

Method	Memory Cost	Training Time
LoRA	43.4 GB	10 h
AdaLoRA	46.7 GB	13 h
IST	36.6 GB	10 h
GREATS	58.1 GB	21 h
GAST (memory-efficient)	51.5 GB	19 h
GAST (computation-efficient)	66.9 GB	11.5 h

Table 6: Computational cost of GAST on llama-7B with weight memory of 14GB and batch size of 16.

maintain the same memory consumption as vanilla PEFT training. Specifically, in the first forward-backward pass, we compute the gradient alignment score for each Linear, and then, after sampling data points for each layer, a second forward-backward pass is performed to select and merge the gradients.

These two implementation strategies allow our algorithm to adapt to various scenarios. Additionally, engineering techniques such as CPU offloading for per-sample gradients and asynchronous gradient aggregation have the potential to further improve the efficiency of the algorithm. In this paper, our focus is on the effectiveness of the algorithm itself rather than its implementation.

B Additional Details and Experiments

B.1 Dataset Statistics

Detailed dataset statistics can be referred to Tab. 5. Note that we trained on Commonsense and Math10K for commonsense reasoning and arithmetic reasoning, respectively. During testing, we evaluated the predefined test sets of each dataset.Meanwhile, we showcase the instructions formats of different datasets in Tab. 8 and Tab. 9.

B.2 Effect of Support Data Used in GAST

Fig 6 (left) illustrates the effect of varying batch size (i.e., K in GAST) of support-set data on training a LLaMA 7B model using the commonsense dataset. We evaluated among five values: [1,2,4,6,8]. Experimental results show that the larger the value of K, the better the performance. However, after K exceeds 4, the improvement becomes marginal. Considering that a larger K also leads to increased computational overhead, we choose $K = 4$ as the default parameter to balance computation and efficiency. Fig. 6 (right) shows the impact of different support set sizes on GAST, ranging from 16 to 10,000. For a fair comparison, we gradually increase the size of the support set used in GAST, but ensure that the training set and the validation set used for evaluation remain unchanged. When the support set size exceeds 100, the benefit of further increasing the set size diminishes significantly. This suggests that a small support set is sufficient to guide relatively accurate gradients. For simplicity and to eliminate the randomness of subset selection, we use the entire training set as the support set by default. When training data contains noise or label errors, GAST may initially be affected if the support set is unrepresentative, as the gradient alignment mechanism could mistakenly treat relevant task features as noise. However, in our experiments in Section B.2, we specifically evaluated using the entire training set as the support set and observed strong robustness in such scenarios. This configuration helps ensure that the gradient alignment process remains stable and does not over-amplify the impact of noisy samples.

B.3 Complexity Analysis

Tab 6 presents the computational costs of different methods for training on Llama-7B under the same setting (weight memory 14GB, batch size 16,

Table 7: Performance comparison on common-sense reasoning benchmarks. Methods marked with * report results from the original papers.

Method	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-E	ARC-C	OBQA	AVG
LoRA	68.9	80.7	77.4	78.1	78.8	77.8	61.3	74.8	74.7
LISA*	–	–	–	–	–	–	–	–	75.3
AdaLoRA*	–	–	–	–	–	–	–	–	76.2
RST	68.3	**82.3**	78.1	76.6	80.1	79.9	63.3	77.8	75.8
IST	68.7	81.7	77.3	82.7	78.7	80.6	62.4	80.0	76.5
GREATS	**69.6**	81.1	78.0	81.0	79.8	79.5	63.8	77.6	76.3
GAST (Random)	68.0	81.4	79.1	**84.8**	79.9	79.7	63.1	79.2	76.9
GAST (Top-K)	50.4	63.7	79.3	77.0	37.7	79.3	63.8	**80.4**	66.4
GAST	68.2	81.6	**79.4**	83.6	**82.2**	**80.4**	**64.7**	79.8	**77.5**

on 80G A100 GPU). Among all methods, LoRA exhibits the lowest memory cost (43.4 GB) and the fastest training time (7 hours), making it the most lightweight option. Data-selective method GREATS, on the other hand, requires significantly more memory (58.1 GB) and has the longest training time (21 hours), suggesting a higher computational overhead. GAST provides two variants to balance efficiency and resource use: the memory-efficient version uses less memory (51.5 GB) but at the cost of longer training (19 hours), while the computation-efficient version reduces training time to 11.5 hours, albeit with a higher memory footprint (66.9 GB). These results demonstrate that GAST enables flexible trade-offs between memory and computation.

B.4 Full results of each subtask in the ablation study

We provide the full results of all the subtasks in the ablation study in Tab 7. * indicate the results obtained from IST.

BoolQ

Please answer the following question with true or false, question: is house tax and property tax are same?
Answer format: true/false
the correct answer is true

PIQA

Please choose the correct solution to the question: Extend life of flowers in vase.
Solution1: Add small amount of coffee in vase.
Solution2: Add small amount of 7UP in vase.
Answer format: solution1/solution2
the correct answer is solution2

SIQA

Please choose the correct answer to the question: Sydney walked past a homeless woman asking for change but did not have any money they could give to her. Sydney felt bad afterwards. How would you describe Sydney?
Answer1: sympathetic Answer2: like a person who was unable to help Answer3: incredulous
Answer format: answer1/answer2/answer3
the correct answer is answer1

HellaSwag

Please choose the correct ending to complete the given sentence: Clean and jerk: A lady walks to a barbell. She bends down and grabs the pole. the lady
Ending1: swings and lands in her arms. Ending2: pulls the barbell forward. Ending3: pulls a rope attached to the barbell. Ending4: stands and lifts the weight over her head.
Answer format: ending1/ending2/ending3/ending4
the correct answer is ending4

WinoGrande

Please choose the correct answer to fill in the blank to complete the given sentence: They were worried the wine would ruin the bed and the blanket, but the _ was't ruined.
Option1: blanket Option2: bed
Answer format: option1/option2
the correct answer is option2

ARC-e

Please choose the correct answer to the question: Which piece of safety equipment is used to keep mold spores from entering the respiratory system?
Answer1: safety goggles Answer2: breathing mask Answer3: rubber gloves Answer4: lead apron
Answer format: answer1/answer2/answer3/answer4
the correct answer is answer2

ARC-c

Please choose the correct answer to the question: The end result in the process of photosynthesis is the production of sugar and oxygen. Which step signals the beginning of photosynthesis?
Answer1: Chemical energy is absorbed through the roots. Answer2: Light energy is converted to chemical energy. Answer3: Chlorophyll in the leaf captures light energy. Answer4: Sunlight is converted into chlorophyll.
Answer format: answer1/answer2/answer3/answer4
the correct answer is answer4

OBQA

Please choose the correct answer to the question: Predators eat
Answer1: lions Answer2: humans Answer3: bunnies Answer4: grass
Answer format: answer1/answer2/answer3/answer4
the correct answer is answer3

Table 8: Instructions format of commonsense reasoning task.

GSM8K

Q: The farm has 30 cows and the zoo has 20 sheep. The zoo has twice as many cows as the farm does. The farm has half as many sheep as the zoo does. How many animals do the farm and zoo have combined?
A: *The farm has 30 cows and the zoo has 20 sheep. The zoo has twice as many cows as the farm, so the zoo has 30 cows. The farm has half as many sheep as the zoo, so the farm has 10 sheep. In total, the farm and zoo have 30 cows + 20 sheep + 10 sheep = 60 animals. The answer is 60.*

Q: Rayna has 60 more dollars than Kassidy, and Kassidy has 3/4 times as much money as Aurelia. If Arelia has $120, how much would each have if they decided to add together their money and share equally among them?
A: *Rayna has 60 more dollars than Kassidy. That means Kassidy has 60 less than Rayna. Kassidy has 3/4 times as much money as Aurelia. That means Aurelia has 4/3 times as much money as Kassidy. The answer is 120.*

AQuA

Q: A rectangular piece of 150 sq m has a length which is 1m more than the 4 times the breadth. What is the perimeter of the piece? Answer Choices: (A) 60 m (B) 61 m (C) 62 m (D) 63 m (E) 64 m
A: *Let the breadth of the rectangular piece be x. Then the length of the piece would be 4x + 1. The area of the piece is 150 sq m, so 4x(4x + 1) = 150. This simplifies to 16x2 + 4x - 150 = 0. Solving this equation, we get x = 5. So the length of the piece is 4x + 1 = 21. The perimeter of the piece is 2(4x + 1) + 2x = 2(21) + 10 = 62 m.The answer is (C).*

Q: Last year, 34 percent of Ace Book Company's sales revenue came from the sale of novels. Of the remaining revenue, 1/3 was from the sale of biographies. The company's revenue from the sale of novels was approximately, how many times its revenue from the sale of biographies? Answer Choices: (A) 1.3 (B) 1.5 (C) 2.1 (D) 2.5 (E) 3.1
A: *34% of the total revenue was from the sale of novels. So the remaining 66% was from the sale of biographies and other books. 66% is equal to 2/3. So the revenue from the sale of novels was approximately 3 times the revenue from the sale of biographies.The answer is (E).*

MAWPS

Q: Zach wants to ride the Ferris wheel , the roller coaster , and the log ride . The Ferris wheel costs 2 tickets , the roller coaster costs 7 tickets and the log ride costs 1 ticket . Zach has 1 ticket . How many more tickets should Zach buy ?
A: *9.0*

Q: In Shannon 's apartment complex , 0.16666666666666666 of the apartments are one - bedroom apartments and 0.3333333333333333 are two - bedroom apartments . What fraction of the apartments are either one- or two - bedroom apartments ?
A: *0.5*

SVAMP

Q: Jack received 9 emails in the morning, 10 emails in the afternoon and 7 emails in the evening. How many more emails did Jack receive in the morning than in the evening?
A: *Jack received 9 emails in the morning, 10 emails in the afternoon and 7 emails in the evening. The difference between the number of emails he received in the morning and the number of emails he received in the evening is 9 - 7 = 2. The answer is 2.*

Q: A mailman has to give 38 pieces of junk mail to each of the 78 blocks. If there are 19 houses on a block. How many pieces of junk mail should he give each house?
A: *The mailman has to give 38 pieces of junk mail to each of the 78 blocks. That means he has to give 78 * 38 = 2964 pieces of junk mail in total. There are 19 houses on a block. So, each house should get 2964 / 19 = 156 pieces of junk mail. The answer is 156.*

Table 9: Instructions format of arithmetic reasoning task.