Low Rank, Lasting Knowledge: Mitigating Catastrophic Forgetting in LoRA Fine-Tuning

Anonymous authors
Paper under double-blind review

Abstract

Low-Rank Adaptation (LoRA) adapts large language models to a target domain (TD) with few trainable parameters, but applying the adapter to every input can degrade performance on out-of-target-domain (OTD) data. The base model itself remains unchanged. We address this non-target interference through DG-LoRA, a sample-level dynamic gating method for the post-hoc conditional activation of a single frozen LoRA adapter. A lightweight multilayer perceptron computes a scalar gate value from an answer-independent routing sequence and uses it to interpolate the output logits of the base and LoRA-enabled models before softmax. This value stays fixed throughout candidate scoring and generation. After training the adapter, we freeze both predictive branches and train only the gate, combining domain classification, target-task and non-target preservation losses. These objectives account for domain membership as well as the effect of adaptation on predictions. Across the evaluated models and tasks, DG-LoRA performed comparably to standard LoRA on TD data while substantially reducing non-target interference on OTD data, including general-domain and other domain-specific tasks. OTD performance generally returned close to the base-model level. The gate added fewer than 10% as many trainable parameters as the LoRA adapter.

1 Introduction

Low-Rank Adaptation (LoRA) significantly reduces the cost of adapting large language models (LLMs) to specific domains and tasks by freezing the base model and learning only a low-rank parameter update(Hu et al., 2022). However, it raises a practical deployment problem: once a domain-specific adapter is attached to a model, it is typically applied to every input. The adapter therefore modifies the model's hidden representations and output distribution even when a request falls outside the original training domain. In such cases, a LoRA-adapted model may perform worse than the original base model(Biderman et al., 2024; Han et al., 2025; Wang et al., 2026a). This behavior is problematic for systems expected to alternate between specialized and general-purpose capabilities(Liu et al., 2024a).

Catastrophic forgetting refers to the loss of previously acquired capabilities following training on a new task(McCloskey & Cohen, 1989; French, 1999). In conventional continual learning, this can occur when updates to shared parameters disrupt behaviour learned on earlier tasks(Guo et al., 2017). LoRA leaves the base-model parameters unchanged. The original model therefore remains available when the adapter is disabled(Hu et al., 2022), even if activating it substantially degrades OTD performance. We study this adapter-dependent degradation as non-target interference. Disabling the adapter restores base-model behaviour, but also removes the TD gains provided by adaptation. The practical challenge is therefore to control the adapter's contribution for each input, retaining useful adaptation while limiting interference on other tasks. This makes input-dependent adapter use a means of addressing the problem without retraining the base model or modifying the trained adapter.

Most work on parameter-efficient fine-tuning (PEFT) has focused on learning effective adapters at reduced cost in computation and storage(Houlsby et al., 2019; Hu et al., 2022; Dettmers et al., 2023). Research on adapter gating and routing has largely considered settings involving different levels, such as sample gating and token routing(Ostapenko et al., 2024; Kong et al., 2024; Han et al., 2025). Neither line of work fully addresses a common deployment scenario: a single LoRA adapter has already been trained for a target domain, but subsequent user requests cannot be assumed to come exclusively from that domain. Applying the adapter indiscriminately may improve performance on

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

target-domain inputs while degrading the model's accuracy elsewhere(Wang et al., 2026a). The system must therefore decide, for each input, whether and how the adapter should be activated.

Retraining the adapter would undermine the purpose of post-hoc deployment, while feature-based domain rules are unlikely to handle semantically related domains, ambiguous task boundaries, or cross-task transfer. We formulate this setting as the post-hoc conditional activation of a single frozen adapter. Importantly, this is not simply a domain-classification problem. Domain membership and adapter utility are separate: some target-domain examples may not benefit from the adapter, whereas some nominally out-of-domain examples may exhibit positive transfer(Huang et al., 2024; Ostapenko et al., 2024). A gate trained only to recover domain boundaries may therefore retain harmful adaptations while suppressing useful ones.

A reasonable gating mechanism should satisfy three constraints. First, its routing decision must not depend on the reference answer. Second, the decision must remain fixed across all candidate answers or autoregressive decoding steps associated with the same input. Third, the controller must leave both the base model and the trained LoRA adapter unchanged.

To meet these requirements, we propose DG-LoRA, a post-hoc dynamic gating method for frozen LoRA adapters. Given a frozen base model M_b and a task adapter A, DG-LoRA predicts a continuous, prompt-level gate from a representation that excludes the reference answer. Before the softmax operation, this gate interpolates between the full output logits of the base and LoRA-enabled branches. The same gate value is reused for all candidate answers or generation steps associated with an input.

Training proceeds in two stages. We first train a task-specific LoRA adapter. We then freeze the base model and adapter and train only the lightweight gate, combining domain classification, target-task and non-target preservation losses. This separates adapter learning from deciding how much the trained adapter should contribute to each input. The output interpolation recovers the behaviour of the base model at $g = 0$ and the LoRA-enabled model at $g = 1$.

Our main contributions are as follows:

- We formulate the post-hoc conditional activation of a single frozen LoRA adapter, distinguishing non-target interference from forgetting caused by changes to base-model parameters. This formulation separates domain membership from adapter utility.
- We introduce DG-LoRA, which computes one answer-independent scalar gate per input to interpolate the output logits of the base and LoRA-enabled models before softmax. The gate remains fixed across candidate answers and generation steps. We train it using domain classification, target-task and non-target preservation losses, while keeping both predictive branches frozen.
- We evaluate DG-LoRA across multiple models and target domains. It performs comparably to standard LoRA on TD data while generally recovering OTD performance to near the base-model level, including on general-domain and other domain-specific tasks. The gate adds fewer than 10% as many trainable parameters as the LoRA adapter.

2 Related Work

2.1 Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT) adapts frozen pretrained models through a small number of trainable parameters. Representative approaches include layer-wise adapters (Houlsby et al., 2019), Prefix-tuning (Li & Liang, 2021), and Prompt-tuning (Lester et al., 2021). As a typical PEFT method, LoRA instead factorizes each update as ($\Delta W = BA$) and trains only the low-rank factors (Hu et al., 2022). LoRA variants focus on improving rank allocation, scalability, and expressiveness through adaptive ranks (Zhang et al., 2023), quantization (Dettmers et al., 2023), and magnitude–direction decomposition (Liu et al., 2024c). Despite preserving the backbone, LoRA can still degrade general capabilities (Biderman et al., 2024) and limit the model's access to world knowledge (Dou et al., 2024). Continual-learning methods mitigate such interference by enforcing orthogonal or weakly interacting LoRA subspaces (Wang et al., 2023; Liang & Li, 2024; Wu et al., 2025; Wang et al., 2025a; 2026b).

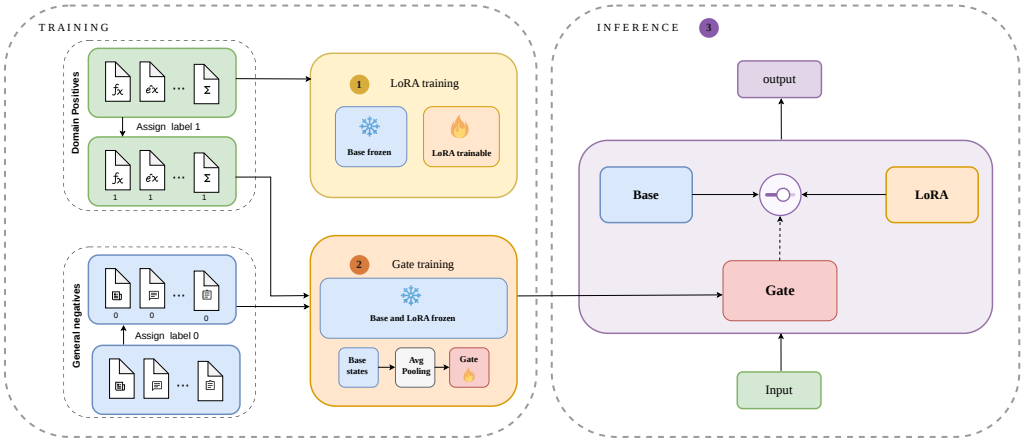


Figure 1: Overview of DG-LoRA. (1) Domain-positive samples train the LoRA adapter while the base model remains frozen. (2) Domain-positive and general-negative samples train the gate using average-pooled base-model states, with both the base model and LoRA adapter frozen. (3) During inference, the input-dependent gate controls the relative contribution of the base-only and LoRA-enabled prediction scores to the final output.

2.2 Token-Level Routing

Many works on mitigating LoRA interference focus on token-level routing. Sparse MoE models route each token to selected experts, using top-1 routing in Switch Transformer (Fedus et al., 2022) and top-2 routing in Mixtral (Jiang et al., 2024a). LoRA-based MoE methods similarly route among low-rank experts. LoRAMoE balances experts specializing in world knowledge and downstream tasks (Dou et al., 2024); MOLE dynamically combines layer-wise LoRA modules (Wu et al., 2024b); and HydraLoRA uses a shared matrix with expert-specific matrices (Tian et al., 2024). Related extensions increase subspace capacity (Wu et al., 2024a), support multimodal tasks (Chen et al., 2024), coordinate competing experts (Lin et al., 2025), or reduce continual-learning interference (Wang et al., 2025b). LoRA-Flow and MeteoRA route among pretrained adapter libraries (Ping et al., 2024; Xu et al., 2025). Unlike these token-level multi-expert methods, DG-LoRA makes one prompt-level decision for a single frozen adapter.

2.3 Sample-Level Routing

Sample-level methods route once per sample. LoRA-Switch shares one routing decision across layers (Kong et al., 2024), while MixLoRA constructs input-dependent low-rank factors (Shen et al., 2024). MoA combines task adapters using domain metadata (Feng et al., 2024); MoLA learns task-conditioned or task-agnostic mixtures (Zhou et al., 2024); and MOELoRA and MoCLE select experts using task IDs or instruction clusters (Liu et al., 2024b; Gou et al., 2024). CLEAR continuously gates a safety adapter (Wang et al., 2026a), whereas Latent-LoRA performs training-free adapter selection from pooled representations (Azghan et al., 2026). Most of these works assume multiple adapters, task identifiers, or domain metadata. DG-LoRA instead learns an answer-independent continuous gate between a frozen base model and one frozen task adapter, optimizing adapter utility rather than domain membership alone.

3 DG-LoRA Model

A LoRA adapter trained for a target task may decrease performance on non-target tasks. To retain target-task gains while preserving the behavior of the base model, DG-LoRA uses **sample-level gating**. Before prediction, the gate computes a scalar value from each sample's routing input to control the contribution of the LoRA-enabled branch. The value is shared across all candidate scores and generation steps for that sample and remains fixed across prediction positions. During gate

training, only the gate parameters are updated; both the base model and the trained LoRA adapter remain frozen.

3.1 Problem Formulation

LoRA adapts a pretrained model through low-rank weight updates (Hu et al., 2022). Consider a linear module with base weight W and input x. LoRA freezes W and trains two low-rank matrices, A and B, which define the update $\Delta W = \frac{\alpha}{r} BA$, where r is the rank and α is the scaling factor. Standard LoRA applies the same effective weight $W + \Delta W$ to every input.

DG-LoRA addresses the inflexibility of standard LoRA with a sample-level gating function $g_\varphi(r(x))$. For an input sample (x, y), let x denote the question context and y the reference information containing the answer. DG-LoRA first constructs the answer-independent routing input $r(x)$ from x and then computes a scalar gate value. This value controls the relative contributions of the base and LoRA-enabled branches. The corresponding effective weight is:

$$W(x) = W_0 + g_\varphi\big(r(x)\big)\Delta W, \tag{1}$$

where $g_\varphi(r(x)) \in (0, 1)$. The learned model parameters remain fixed across samples; only the sample-dependent mixing coefficient and resulting predictions change. For convenience, we denote the model equipped with this gate by M_φ.

For a sample (x, y), let $(x, y_{<t})$ denote the input context at prediction step t. The resulting logits are:

$$z_t = M_{\varphi,x}\left(x, y_{<t}\right), \tag{2}$$

With all other settings fixed, gate values of 0 and 1 recover the base model and the fully LoRA-enabled model, respectively. Intermediate values interpolate between the two predictive branches.

3.2 Prompt-Level Gate

For an input sample x, let $r(x)$ denote the routing sequence available to the gate; it excludes the reference answer to be predicted. The gate uses $r(x)$ to compute the mixing coefficient for that input.

To obtain the gate value, we pass $r(x)$ through the base model alone and extract the last-layer hidden states $H_b(r(x)) = (h_1, \ldots, h_T)$, where T is the sequence length, $h_j \in \mathbb{R}^d$, and d is the model's hidden dimension. We then apply masked mean pooling over the hidden states at non-padding positions, converting variable-length inputs into fixed-dimensional gate features. This operation introduces no trainable parameters. Normalizing by the number of valid tokens also prevents longer sequences from accumulating larger feature magnitudes, as would occur with sum pooling.

$$\overline{h}\big(r(x)\big) = \left(\sum_{j=1}^{T} m_j h_j\right) \Big/ \max\left(1, \sum_{j=1}^{T} m_j\right), \qquad m_j \in \{0, 1\}. \tag{3}$$

Here, $m_j = 1$ marks a valid token, whereas $m_j = 0$ marks a padding position. Masked mean pooling converts variable-length inputs into fixed-dimensional vectors from which the gate computes the mixing coefficient.

The gate is a two-layer MLP with hidden dimension h. Given the pooled representation, its nonlinear hidden layer captures the relation between the input features and the relative contributions of the base and LoRA-enabled branches. The output layer maps this representation to a scalar logit, which is passed through a sigmoid function to obtain the gate value:

$$s_\varphi\big(r(x)\big) = W_2\left(\mathrm{GELU}\left(W_1\overline{h}\big(r(x)\big) + b_1\right)\right) + b_2, \tag{4}$$

$$g_\varphi\big(r(x)\big) = \sigma\left(s_\varphi\big(r(x)\big)\right) \in (0, 1). \tag{5}$$

Here, W_1, b_1, W_2, and b_2 are trainable parameters. The sigmoid function constrains the gate value to $(0, 1)$, allowing gradient-based optimization to adjust the contribution of the LoRA-enabled branch. During training, we apply dropout with probability 0.1 to reduce the gate's reliance on training-specific features. Dropout is disabled at inference.

The gate value is computed once from the routing input and reused for all candidate scores and subsequent generation steps associated with that input. It remains independent of candidate content and the evolving generation history, ensuring that the two branches use the same mixing coefficient throughout prediction.

3.3 Gate Training Objective

We train the gate after obtaining the LoRA adapter. During this stage, both the base model and the adapter remain frozen; only the gate parameters are optimized.

Gate Training Objective. The gate must infer how strongly an input is associated with the target task from its routing information and adjust the contribution of the LoRA-enabled branch accordingly. Domain membership alone, however, does not determine the appropriate gate value. A target-domain sample may not require full adapter activation, while LoRA may affect different non-target samples to different degrees. We therefore combine three objectives: a domain classification loss, a target-task loss, and a non-target preservation loss. These provide domain supervision, target-prediction supervision, and a constraint that preserves the behavior of the base model, respectively.

The two sample groups serve different purposes during training. For target-domain samples, we use the correct answers as supervision and minimize negative log-likelihood to learn mixing coefficients that improve task prediction. For non-target samples, we treat the base model as the reference and use KL divergence to constrain changes in the predictive distribution. The former promotes target-task adaptation; the latter preserves base-model behavior.

For a training sample (x_i, y_i), let d_i denote its domain label, where $d_i = 1$ indicates the target domain and $d_i = 0$ a non-target domain. The gate computes $g_i = g_\varphi(r(x_i))$ from the routing input and applies this value to every position included in the prediction loss for that sample. Let I_+ and I_- denote the sets of valid token positions used to compute the losses for target and non-target samples, respectively.

Domain Classification Loss (DC): Because the target-task LoRA adapter is trained primarily on target-domain data, domain membership provides an initial signal for determining its contribution. We train the gate with binary cross-entropy, encouraging larger gate values for target-domain samples and smaller values for non-target samples:

$$\mathcal{L}_{\mathrm{DC}} = -\frac{1}{\mathcal{B}} \sum_{(x,y)\in\mathcal{B}} d(x) \log g_\varphi\big(r(x)\big) + \big(1 - d(x)\big) \log\big(1 - g_\varphi\big(r(x)\big)\big) \tag{6}$$

This loss teaches the gate to identify target-domain features from the routing input and provides explicit supervision for the mixing coefficient. However, domain labels do not indicate whether activating the adapter improves a particular prediction. The domain classification loss alone is therefore insufficient to determine the final gate value.

Target-Task Loss (TT): Domain labels indicate whether an input belongs to the target task, but they do not determine the appropriate mixing coefficient. Domain supervision alone pushes the gate values of target-domain samples toward 1, even when doing so does not improve prediction. We therefore introduce a target-task loss that adjusts the mixing coefficient according to the probability assigned by the gated model to the correct token:

$$\mathcal{L}_{\mathrm{TT}} = -\frac{1}{N_{\mathrm{tar}}} \sum_{(x,y)\in\mathcal{B}_{\mathrm{tar}}} \sum_{t\in T(x,y)} \log p_{\mathrm{DG},t}\left(y_t \mid x, y_{<t}\right) \tag{7}$$

Here, $\mathcal{B}_{\mathrm{tar}}$ denotes the set of target-domain samples in the training batch, and $T(x, y)$ denotes the set of valid token positions included in the loss for sample (x, y). We define $N_{\mathrm{tar}} = \sum_{(x,y)\in\mathcal{B}_{\mathrm{tar}}} |T(x, y)|$ as the total number of valid tokens. y_t is the current supervised token, and $y_{<t}$ is the preceding token sequence provided by the training text.

As shown in the equation above, $\mathcal{L}_{\text{TT}}$ decreases as $p_{\text{DG,t}}$ assigns greater probability to the correct token, indicating better target-task performance. Because both the base model and the LoRA adapter are frozen, this loss can affect predictions only by updating the gate. If increasing g_i reduces $\mathcal{L}_{\text{TT}}$, optimization pushes the gate value upward; otherwise, it pushes the value downward. The gate therefore adjusts its mixing coefficient using both domain labels and observed target-task performance.

Non-Target Preservation Loss (NTP): For non-target samples, the LoRA update may alter predictions made by the base model. Reducing the gate value through domain supervision alone does not directly constrain this change. We therefore use the base model's predictive distribution as a reference and compute the average $\text{KL}(p_b|p_{\text{DG}})$ over valid token positions in non-target samples:

$$\mathcal{L}_{\text{NTP}} = \frac{1}{N_{\text{non}}} \sum_{(x,y)\in\mathcal{B}_{\text{non}}} \sum_{t\in T(x,y)} D_{\text{KL}}\left(p_{b,t}(\cdot \mid x, y_{<t}) \,\|\, p_{\text{DG},t}(\cdot \mid x, y_{<t})\right) \tag{8}$$

Here, $\mathcal{B}_{\text{non}}$ denotes the set of non-target samples in the batch, and $T(x, y)$ denotes the valid token positions included in the loss for sample (x, y). The total number of valid tokens is $N_{\text{non}} = \sum_{(x,y)\in\mathcal{B}_{\text{non}}}|T(x, y)|$. p_b and p_{DG} denote the predictive distributions of the base and gated models, respectively, with p_{DG} obtained by applying softmax to the gated logits. Both distributions are computed from the same input and preceding token sequence, and the KL divergence is summed over the full vocabulary.

The loss decreases as the two predictive distributions become closer. Because the base model and LoRA adapter are frozen, optimization acts only on the gate, adjusting the mixing coefficient so that predictions on non-target samples remain close to the original behavior of the base model.

Joint Objective: The three losses capture domain classification, target-task prediction, and non-target behavior preservation, respectively. Each is differentiable with respect to the gate parameters through the gate value. We combine them in a single objective with equal weights:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{DC}} + \mathcal{L}_{\text{TT}} + \mathcal{L}_{\text{NTP}} \tag{9}$$

During optimization, gradients from the three losses jointly determine the gate update. Domain supervision provides an initial preference for the mixing coefficient, the target-task loss adjusts it according to the probability assigned to the correct token, and the preservation loss limits predictive deviation on non-target samples.

4 Experiments and Results

4.1 Experimental Setup

Models and Datasets Three backbone models of different sizes are evaluated: Llama3.2-1B, Llama3.2-3B-Instruct, and Llama3-8B. They are fine-tuned on a target domain (TD) and then evaluated on other domains (out-of-target-domain, OTD). We select five well-known benchmark datasets. Among them, GSM8K (Cobbe et al., 2021) and MedMCQA (Pal et al., 2022) are domain-specific, covering mathematics and medicine, respectively; BIG-bench Hard (BBH) (Suzgun et al., 2022), HellaSwag (HS) (Zellers et al., 2019), and WikiText-2 (Wiki) (Merity et al., 2017) cover general domains. These datasets span a variety of NLP tasks, such as mathematics, logical reasoning, and question answering. To train the gate, we label the data used to fine-tune LoRA as positive and select an equal number of negative examples. These negative examples come from MMLU (Hendrycks et al., 2021), ARC-Challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), UltraChat (Ding et al., 2023), and Alpaca (Taori et al., 2023). The proportions from these sources are 20%, 10%, 20%, 25%, and 25%, respectively. Examples from the target domain are excluded. The details are shown in the appendix.

Baselines The base-model baseline is evaluated on the test data without fine-tuning. The LoRA baseline uses standard LoRA fine-tuning. We run each experiment with three different seeds and report the average.

Table 1: Results after fine-tuning on domain-specific tasks and evaluating on other domain-specific and general-domain tasks. Parenthesized results use GSM8K as the target domain. Med. abbreviates MedMCQA. Accuracy values are percentages[a].

Model	Method	Target domain	Out-of-target domain (GSM8K)			
		Med. ↑ (GSM8K↑)	GSM8K ↑	HS ↑	BBH ↑	Wiki ↓
Llama3.2-1B	base	35.45 (4.32)	4.32	59.44	24.47 (25.25)	2.66 (2.66)
	Lora	37.97 (21.71)	0.00	26.85	14.34 (21.71)	16.49 (2.76)
	Ours	37.93 (21.63)	4.4	59.44	24.24 (25.34)	2.66 (2.66)
Llama3.2-3B-Instruct	base	46.67 (40.03)	11.68	61.91	38.98 (38.98)	2.71 (2.70)
	Lora	50.21 (56.66)	0.00	59.95	20.15 (24.21)	3.82 (2.71)
	Ours	50.19 (56.68)	10.61	61.91	38.92 (40.00)	2.71 (2.70)
Llama3-8B	base	56.83 (23.81)	23.58	74.51	35.08 (36.45)	2.20 (2.20)
	Lora	59.57 (52.99)	0.00	26.26	15.92 (20.39)	13.51 (2.26)
	Ours	59.54 (52.64)	23.12	74.51	35.07 (36.32)	2.24 (2.20)

[a] The performance of the base model varies slightly on the same data.

Metrics We use zero-shot evaluation for all tasks. Negative log-likelihood (NLL) is used for Wiki, and accuracy is used for the other datasets.

Implementation Details Training proceeds in two stages. In the first stage, the base model is frozen, and the LoRA adapter is trained only on the target task. We use a LoRA rank of $r = 8$, a scaling factor of $\alpha = 16$, and a dropout rate of 0.05. The adapter is optimized with AdamW at a learning rate of 2×10^{-4} for up to 3 epochs. The checkpoint with the lowest average per-token negative log-likelihood on the internal validation set is selected for the second stage.

In the second stage, both the base model and the selected LoRA adapter are frozen, and only the gate is trained. The gate applies masked mean pooling to the representations of valid prompt tokens from the final layer of the base model, followed by a two-layer MLP with dimensions $d \to 256 \to 1$. The MLP uses GELU activation, a dropout rate of 0.1, and a sigmoid output layer to produce a scalar gate value. At inference, this value interpolates the base and LoRA-enabled logits before softmax and is shared across all candidate answers or subsequent generation steps for the same input. The gate is trained with three equally weighted objectives: domain-classification loss, target-task loss, and non-target preservation loss. We optimize the gate with AdamW at a learning rate of 10^{-3} for up to 10 epochs. The number of gate parameters depends on the hidden dimension d of the base model.

4.2 Main results

We first examine how target-domain adaptation affects performance on general-domain and other specific-domain data. As shown in Table 1, DG-LoRA consistently matches standard LoRA on the target domain while substantially outperforming it on both types of out-of-target-domain data. On these out-of-target domains, DG-LoRA nearly recovers the performance of the base model. Overall, it performs close to the stronger of standard LoRA and the base model across domains.

Additionally, we fine-tune base models on HellaSwag (HS), a general-domain task, and then evaluate generalization on general-domain and domain-specific benchmarks. Table 2 lists the results. Again, across the three models, our method performs nearly as well as the better of the base model and LoRA on target- and out-of-target-domain tasks.

4.3 Ablation Study

Our loss function consists of a domain classification loss (DC), a target-task loss (TT), and a non-target preservation loss (NTP). We compare four combinations to investigate the influence of the different losses. Table 3 lists the results. The baseline uses only DC. Adding TT improves target-domain performance. On out-of-target-domain data, accuracy on ARC is also higher than the baseline, but the Router score is slightly lower. Adding NTP to the baseline improves only target-domain accuracy. Using all three losses ties for the best BBH accuracy and achieves the highest ARC-Challenge accuracy and Router-AUROC. No combination improves on the baseline for Wiki.

Table 2: Results after fine-tuning on HellaSwag and evaluating on target- and out-of-target-domain tasks.

Model	Method	Target domain	Out-of-target domain		
		HS ↑	GSM8K ↑	BBH ↑	Wiki ↓
Llama3.2-1B	base	59.44	4.32	24.25	2.66
	Lora	64.18	2.65	19.79	2.89
	Ours	64.18	4.40	24.44	2.66
Llama3.2-3B-Instruct	base	61.90	11.14	38.98	2.71
	Lora	72.34	2.15	20.15	2.76
	Ours	72.33	11.37	38.92	2.71
Llama3-8B	base	74.46	23.81	35.43	2.20
	Lora	79.49	4.50	32.63	2.45
	Ours	79.48	23.81	35.26	2.20

Table 3: Ablation study of the three components of the training objective. BBH is the target domain and Llama3.2-1B is the backbone model.

DC	TT	NTP	BBH ↑	ARC-Challenge ↑	Router-AUROC ↑	Wiki ↓
✓	✗	✗	53.17	42.35	98.09	2.66
✓	✓	✗	53.22	42.41	98.05	2.66
✓	✗	✓	53.22	41.92	97.05	2.66
✓	✓	✓	53.22	42.72	98.11	2.66

4.4 Influence of Other Factors

Negative example selection We tested several mixtures of negative examples for gate training, including one drawn from UltraChat, Alpaca, and BoolQ in proportions of 35%, 35%, and 30%, respectively. This mixture yielded results similar to those obtained with the five-dataset mixture used in the main experiments.

Model family To verify that our method is model-agnostic, we fine-tune Qwen3.5-4B with GSM8K as the target domain and then evaluate its performance on BBH. On GSM8K, DG-LoRA achieves 56.86, while LoRA and the base model achieve 57.16 and 2.88, respectively. On BBH, DG-LoRA achieves 51.99, while LoRA and the base model achieve 15.39 and 51.92, respectively. Again, our method achieves performance comparable to or better than that of the base model and LoRA.

Evaluation on a Safety-Prompt Dataset To further investigate our gate mechanism on domain-specific data, we evaluate it on a safety-prompt benchmark. Wang et al. (2026a) fine-tuned Gemma-2-2B-it and Llama-3-8B-Instruct on the WildJailbreak training split (Jiang et al., 2024b) and then tested them on GSM8K and MMLU. The dataset contains 261,599 prompts: 132,778 unsafe and 128,781 safe. Using their dataset, our method outperforms CLEAR in two comparisons and underperforms it in two. When we replace the safe prompts with our general-domain data, our method outperforms CLEAR in all four comparisons. Figure 2 shows the results.

4.5 Additional Parameters and Latency

The gate module requires relatively few trainable parameters: fewer than 10% as many as the corresponding LoRA adapter. The proportions are 9.31%, 6.47%, and 5.01% for the 1B, 3B, and 8B models, respectively. The number of additional parameters is $256d + 513$, where d is the hidden dimension. We observe negligible additional inference latency.

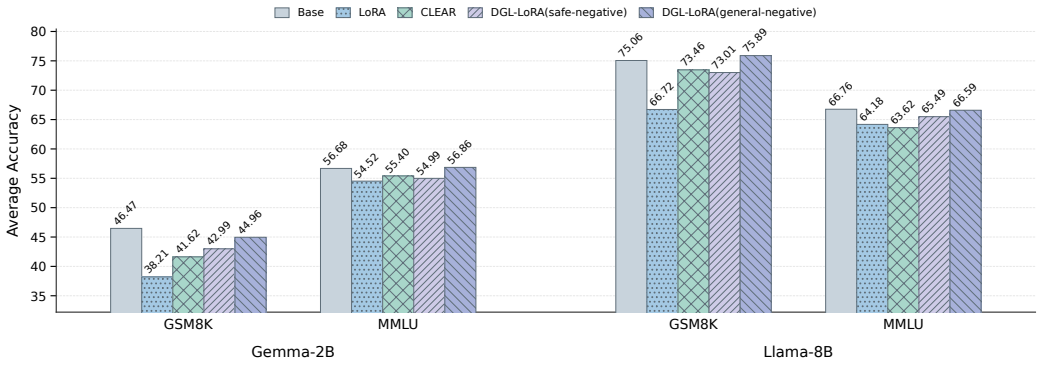


Figure 2: Evaluation on a safety-prompt dataset.

5 Conclusion and Future Work

We have presented DG-LoRA, a sample-level gating method for the post-hoc conditional activation of a single frozen LoRA adapter. A lightweight MLP computes a scalar gate value from an answer-independent routing sequence, then uses it to interpolate the output logits of the base and LoRA-enabled models before softmax. This value stays fixed throughout candidate scoring and generation. We train only the gate. The domain classification, target-task and non-target preservation losses jointly guide its decisions, while the base model and adapter remain unchanged.

In our experiments, DG-LoRA performed comparably to standard LoRA on TD data and substantially reduced non-target interference on OTD data, covering both general-domain and other domain-specific tasks. With the MedMCQA adapter, it brought GSM8K performance back to near the base-model level across the evaluated Llama model sizes, while MedMCQA accuracy differed from standard LoRA by at most 0.04 percentage points. The gate required fewer than 10% as many trainable parameters as the adapter.

The results suggest that retaining target-domain gains need not require accepting the non-target interference caused by always activating an adapter. Learning when to use a trained adapter offers a practical means of managing this trade-off. Domain membership alone, however, cannot determine adapter utility. Our conclusions are limited to the evaluated tasks, a single frozen adapter and curated gate-training data. Further work should examine ambiguous domain boundaries and positive cross-task transfer, where matching base-model behaviour on OTD inputs may suppress useful adaptation. Extending this approach to multiple frozen adapters is another direction.

References

Reza Rahimi Azghan, Gautham Krishna Gudur, Giulia Pedrielli, Pavan Turaga, and Hassan Ghasemzadeh. Latent-LoRA: Compact latent-space adapters with gradient-free routing for continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2026*. Association for Computational Linguistics, 2026.

Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John P. Cunningham. LoRA learns less and forgets less. *Transactions on Machine Learning Research (TMLR)*, 2024. URL `https://openreview.net/forum?id=aloEru2qCG`.

Zeren Chen, Ziqin Wang, Zhen Wang, Huayang Liu, Zhenfei Yin, Si Liu, Lu Sheng, Wanli Ouyang, Yu Qiao, and Jing Shao. Octavius: Mitigating task interference in MLLMs via LoRA-MoE. In *International Conference on Learning Representations (ICLR)*, 2024.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*, pp. 2924–2936, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, 2023.

Ning Ding et al. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023.

Shihan Dou, Enyu Zhou, Yanpeng Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, et al. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 1932–1945. Association for Computational Linguistics, 2024.

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

Wenfeng Feng, Chuzhan Hao, Yuewei Zhang, Yu Han, and Hao Wang. Mixture-of-LoRAs: An efficient multitask tuning for large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING)*, pp. 11371–11380. ELRA and ICCL, 2024.

Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3 (4):128–135, April 1999. doi: 10.1016/S1364-6613(99)01294-2. URL `https://doi.org/10.1016/S1364-6613(99)01294-2`.

Yunhao Gou, Zhili Liu, Kai Chen, Lanqing Hong, Hang Xu, Aoxue Li, Dit-Yan Yeung, James T. Kwok, and Yu Zhang. MoCLE: Mixture of cluster-conditional LoRA experts for vision-language instruction tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 2024.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1321–1330, 2017.

Jiayi Han, Liang Du, Hongwei Du, Xiangguo Zhou, Yiwen Wu, Yuanfang Zhang, Weibo Zheng, and Donghong Han. SLIM: Let LLM learn more and forget less with soft LoRA and identity mixture. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4792–4804. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-long.246. URL `https://aclanthology.org/2025.naacl-long.246/`.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 2019. URL `https://proceedings.mlr.press/v97/houlsby19a.html`.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=nZeVKeeFYf9`.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. LoRAHub: Efficient cross-task generalization via dynamic LoRA composition. In *First Conference on Language Modeling (COLM)*, 2024.

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, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024a.

Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, and Nouha Dziri. WildTeaming at Scale: From In-the-Wild Jailbreaks to (Adversarially) Safer Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b.

Rui Kong, Qiyang Li, Xinyu Fang, Qingtian Feng, Qingfeng He, Yazhu Dong, Weijun Wang, Yuanchun Li, Linghe Kong, and Yunxin Liu. LoRA-Switch: Boosting the efficiency of dynamic LLM adapters via system-algorithm co-design. *arXiv preprint arXiv:2405.17741*, 2024.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3045–3059. Association for Computational Linguistics, 2021.

Xiang Lisa Li and Percy Liang. 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 (ACL-IJCNLP)*, pp. 4582–4597. Association for Computational Linguistics, 2021.

Yan-Shuo Liang and Wu-Jun Li. InfLoRA: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23638–23647, Seattle, WA, USA, June 2024. IEEE. doi: 10.1109/CVPR52733.2024.02231. URL https://openaccess.thecvf.com/content/CVPR2024/html/Liang_InfLoRA_Interference-Free_Low-Rank_Adaptation_for_Continual_Learning_CVPR_2024_paper.html.

Tianwei Lin, Jiang Liu, Wenqiao Zhang, Yang Dai, Haoyuan Li, Zhelun Yu, Wanggui He, Juncheng Li, Jiannan Guo, Hao Jiang, Siliang Tang, and Yueting Zhuang. TeamLoRA: Boosting low-rank adaptation with expert collaboration and competition. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 13622–13637. Association for Computational Linguistics, 2025.

Chengyuan Liu, Yangyang Kang, Shihang Wang, Lizhi Qing, Fubang Zhao, Chao Wu, Changlong Sun, Kun Kuang, and Fei Wu. More than catastrophic forgetting: Integrating general capabilities for domain-specific LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7531–7548, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.429. URL https://aclanthology.org/2024.emnlp-main.429/.

Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. When MOE meets LLMs: Parameter efficient fine-tuning for multi-task medical applications. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 1104–1114. ACM, 2024b.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121. PMLR, 21–27 Jul 2024c. URL https://proceedings.mlr.press/v235/liu24bn.html.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower (ed.), *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Academic Press, 1989. doi: 10.1016/S0079-7421(08)60536-8. URL https://doi.org/10.1016/S0079-7421(08)60536-8.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

Oleksiy Ostapenko, Zhan Su, Edoardo Ponti, Laurent Charlin, Nicolas Le Roux, Lucas Caccia, and Alessandro Sordoni. Towards modular LLMs by building and reusing a library of LoRAs. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 38885–38904. PMLR, 2024. URL https://proceedings.mlr.press/v235/ostapenko24a.html.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. MedMCQA: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of the Conference on Health, Inference, and Learning*, pp. 248–260, 2022.

Bowen Ping, Hanqing Wang, Maosong Sun, Shuo Wang, Xu Han, Yun Chen, and Zhiyuan Liu. LoRA-Flow: Dynamic LoRA fusion for large language models in generative tasks. *arXiv preprint arXiv:2402.11455*, 2024.

Ying Shen, Zhiyang Xu, Qifan Wang, Yu Cheng, Wenpeng Yin, and Lifu Huang. Multimodal instruction tuning with conditional mixture of LoRA. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 637–648, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.38. URL https://aclanthology.org/2024.acl-long.38/.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, et al. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, et al. Stanford alpaca: An instruction-following llama model. https://crfm.stanford.edu/2023/03/13/alpaca.html, 2023.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. HydraLoRA: An asymmetric LoRA architecture for efficient fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 9565–9584, 2024.

Chengxiao Wang, Enyi Jiang, Xiaojing Liao, and Sanmi Koyejo. CLEAR: Continuous latent adapter routing for utility-preserving LLM safety alignment. *arXiv preprint arXiv:2608.21278*, 2026a.

Chenxu Wang, Yilin Lyu, Zicheng Sun, and Liping Jing. Continual gradient low-rank projection fine-tuning for LLMs. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14815–14829, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.721. URL https://aclanthology.org/2025.acl-long.721/.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10658–10671. Association for Computational Linguistics, 2023.

Yitong Wang, Xue Han, WenChun Gao, Qian Hu, Jiahui Wang, Ziqing Wang, Lijun Mei, and Junlan Feng. Soft orthogonal low-rank adaptation for knowledge sharing in large language model continual learning. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18472–18487, San Diego, California, United States, July 2026b. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.842. URL https://aclanthology.org/2026.acl-long.842/.

Zitao Wang, Xinyi Wang, and Wei Hu. Mixture of LoRA experts for continual information extraction with LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 13324–13339. Association for Computational Linguistics, 2025b.

Taiqiang Wu, Jiahao Wang, Zhe Zhao, and Ngai Wong. Mixture-of-subspaces in low-rank adaptation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7880–7899. Association for Computational Linguistics, 2024a.

Xun Wu, Shaohan Huang, and Furu Wei. Mixture of LoRA experts. In *International Conference on Learning Representations (ICLR)*, 2024b.

Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. SD-LoRA: Scalable decoupled low-rank adaptation for continual learning. In *International Conference on Learning Representations (ICLR)*, 2025.

Jingwei Xu, Junyu Lai, and Yunpeng Huang. MeteoRA: Multiple-tasks embedded LoRA for large language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

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

Yuhang Zhou, Zihua Zhao, Siyuan Du, Haolin Li, Jiangchao Yao, Ya Zhang, and Yanfeng Wang. Exploring training on heterogeneous data with mixture of low-rank adapters. In *International Conference on Machine Learning (ICML)*, 2024.

A Appendix 1: Details of Gate-Training Data

We use seed 42 throughout. This seed controls both the dataset splits and gate-sample construction for all three domains.

The official GSM8K training set contains 7,473 examples, which we divide into 6,726 for training and 747 for validation. LoRA uses this split for training and checkpoint selection. For the gate, all examples in the two partitions serve as positive samples, and we pair them with 6,726 negative training samples and 747 negative validation samples. Evaluation remains fully separate. We report target-domain performance on the official GSM8K test set of 1,319 examples, none of which appears during LoRA or gate training.

HellaSwag provides 39,905 official training examples. We split them into 35,915 training and 3,990 validation examples, then sample without replacement 6,726 positive training examples and 747 positive validation examples for the gate. Each partition receives the same number of negative examples. We evaluate target-task performance on the official HellaSwag validation set, which contains 10,042 examples.

The official MedMCQA training set is larger, with 182,822 examples. We divide it into 164,540 training and 18,282 validation examples. From the corresponding partitions, we sample without replacement 6,726 positive training examples and 747 positive validation examples, again pairing each set with an equal number of negatives. Target-domain performance is measured on the 4,183 examples in the official MedMCQA validation set.

Every gate training and validation set therefore has a 1:1 positive-to-negative ratio. The positive gate-validation examples comprise either all or a subset of the internal validation examples used to select the LoRA checkpoint.

For all three target domains, negative samples for gate training come from MMLU, ARC-Challenge, BoolQ, UltraChat, and Alpaca, with sampling proportions of 20%, 10%, 20%, 25%, and 25%, respectively. We exclude any samples related to the target domain.

The non-target evaluation sets vary by target domain. For GSM8K, we evaluate on BBH and WikiText-2. For HellaSwag, we use BBH, GSM8K, and WikiText-2. The MedMCQA experiments cover a broader set: MedQA, PubMedQA, GSM8K, HellaSwag, BBH, and WikiText-2.

For CLEAR, we retain the experimental setup of the original paper. During gate training, however, we supplement its original negative-sample dataset with the same general-domain negative pool constructed from the five datasets above. No evaluation dataset is used to sample negative examples for gate training in any experiment. We report the mean or median across three random seeds, as appropriate.

B Appendix 2: Additional Experimental Results

B.1 Performance on BBH

Table 4: Out-of-Domain Performance Recovery under BBH Adaptation on Llama-3.2-1B

Dataset / metric	Base	LoRA	DG-LoRA
BBH	19.43	53.14	53.22
GSM8K	4.40	2.58	4.27
ARC-Challenge	42.24	36.41	42.72
Wiki ↓	2.6608	2.7354	2.6614

To complement the GSM8K-targeted experiment in the main paper, we reverse the domain assignment and adapt Llama-3.2-1B to BBH, using GSM8K as the primary out-of-domain evaluation task. We also report results on ARC-Challenge and Wiki to examine whether the effect extends beyond GSM8K. As shown in Table 4, standard LoRA raises the BBH score from 19.43 to 53.14. This gain is accompanied by lower scores on GSM8K (4.40 to 2.58) and ARC-Challenge (42.24 to 36.41), while the Wiki metric

increases from 2.6608 to 2.7354, where lower is better. Thus, adaptation to BBH improves the target task but degrades all three reported out-of-domain measures under standard LoRA.

DG-LoRA achieves a BBH score of 53.22, matching the target-domain improvement of standard LoRA while recovering much of the out-of-domain performance. On GSM8K, it restores 1.69 of the 1.82 points lost under LoRA, reaching 4.27, only 0.13 points below the base model. Its ARC-Challenge score rises to 42.72, exceeding the base model's 42.24, and its Wiki metric returns to 2.6614, close to the base value of 2.6608. Since these improvements occur without a reduction in BBH performance, the results suggest that DG-LoRA mitigates the out-of-domain degradation associated with LoRA rather than trading target-domain accuracy for performance elsewhere. Together with the GSM8K-targeted experiment, this reversed setting provides evidence that the effect is not limited to one choice of target domain.

C Use of Large language models

LLMs are used exclusively for minor writing assistance, for example, to enhance grammar. correct typos, and improve sentence clarity.