SCALE: Upscaled Continual Learning of Large Language Models

Jin-woo Lee*, **Junhwa Choi***, **Bongkyu Hwang, Jinho Choo, Bogun Kim, JeongSeon Yi, Joonseok Lee, DongYoung Jung, Jaeseon Park, Kyoungwon Park, Suk-hoon Jung**†
Samsung SDS

Abstract

We revisit continual pre-training for large language models and argue that progress now depends less on scaling parameters than on scaling the right structure. We introduce SCALE, a width upscaling architecture that inserts lightweight expansions into linear modules while freezing all pre-trained parameters, preserving residual and attention topologies and increasing capacity without perturbing the base model's original functionality. SCALE follows two principles: *Persistent Preservation*, which maintains the base model's behavior via preservation-oriented initialization and freezing of the pre-trained weights, and *Collaborative Adaptation*, which trains only selected expansion components to acquire new knowledge with minimal interference. We instantiate these ideas as SCALE-Preserve (preservation-first), SCALE-Adapt (adaptation-first), and SCALE-Route, an optional routing extension that performs token-level routing between preservation and adaptation heads. On a controlled synthetic biography benchmark, SCALE reduces the severe forgetting seen in depth expansion while still learning new knowledge. In continual pre-training on a Korean corpus, SCALE variants forget less on English evaluations and achieve competitive gains on Korean benchmarks, yielding the best overall stability-plasticity trade-off. We further analyze when preservation holds provably and why combining preservation and adaptation stabilizes optimization relative to standard continual learning.

1 Introduction

The era of effortless gains from brute-force scaling of large language models (LLMs) is nearing its end. Recent discussions suggest that further progress will come from scaling the right structure, not merely parameters or data, while preserving acquired knowledge (Sutskever, 2024; LeCun, 2025).

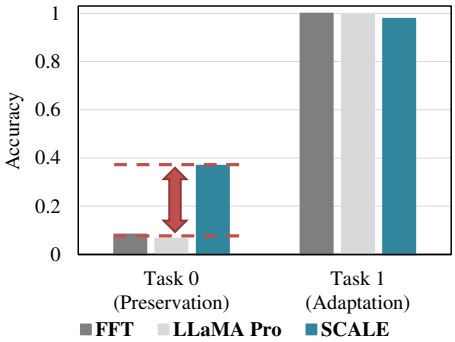


Figure 1: **Continual biography learning.**

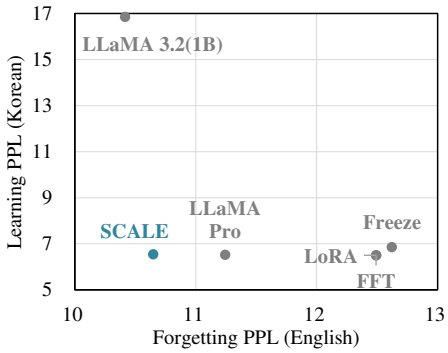


Figure 2: **Performance landscape.**

This shifts attention to *strategic architectural expansion* for continual pre-training (CPT): adding capacity while preserving the pre-trained knowledge base. Classical continual learning (CL), such as regularization, replay, and parameter isolation, improves retention but typically does not add new capacity (Kirkpatrick et al., 2017; Rolnick et al., 2019). By contrast, function-preserving transformations such as Net2Net (Chen et al., 2015) show that expansion can increase capacity without disrupting the original function; however, depth upscaling such as LLaMA Pro (Wu et al., 2024) can still perturb representations and trigger forgetting during CPT.

We propose SCALE (*upScaled ContinuAl LEarning*), a width-upscaling architecture that in-

*Equal contribution.
†Correspondence: sukhoon.jung@samsung.com

Findings of the Association for Computational Linguistics: ACL 2026, pages 41006–41020

serts lightweight expansions into linear modules while freezing all pre-trained parameters. SCALE increases capacity in decoder-style LLMs without changing residual topology or attention structure. Figure 1 shows that depth-upscaled LLaMA Pro suffers severe forgetting on *continual biography* learning , whereas width-upscaled SCALE preserves prior knowledge significantly better while adapting to new knowledge. In CPT on a Korean dataset, Figure 2 shows lower English-forgetting perplexity and competitive Korean learning perplexity across baselines, including LLaMA Pro and Freeze (Zheng et al., 2025), indicating a superior stability-plasticity balance.

SCALE is built on two complementary principles: ① *Persistent Preservation* and ② *Collaborative Adaptation*. *Persistent Preservation* maintains the original function during training via preservation-oriented initialization and freezing patterns. *Collaborative Adaptation* trains only selected expansion blocks (e.g., upper layers or specific modules) to acquire new domain knowledge with minimal interference. Empirically, preservation-first settings strongly resist forgetting, while collaborative settings improve adaptability; together they define a controllable stability-plasticity frontier.

Based on these principles, we introduce SCALE-Preserve (preservation-first), SCALE-Adapt (adaptation-first), and SCALE-Route, a routing extension that performs token-level routing between preservation and adaptation paths. Because adaptation can override preservation, SCALE-Route exposes both behaviors in a single forward pass and selects the more relevant logits per token. We also derive a tighter convergence bound for routing-based CL than for standard CL, supporting the observed stability-plasticity gains.

Key Contributions

1. *Width Upscaling Architecture.* We propose SCALE, which freezes all pre-trained parameters and adds lightweight expansions inside linear modules, preserving base knowledge while adding adaptive capacity (Section 3).

2. *Principles with Evidence.* We formalize *Persistent Preservation* and *Collaborative Adaptation*, supported by theory and preliminary studies (Section 3).

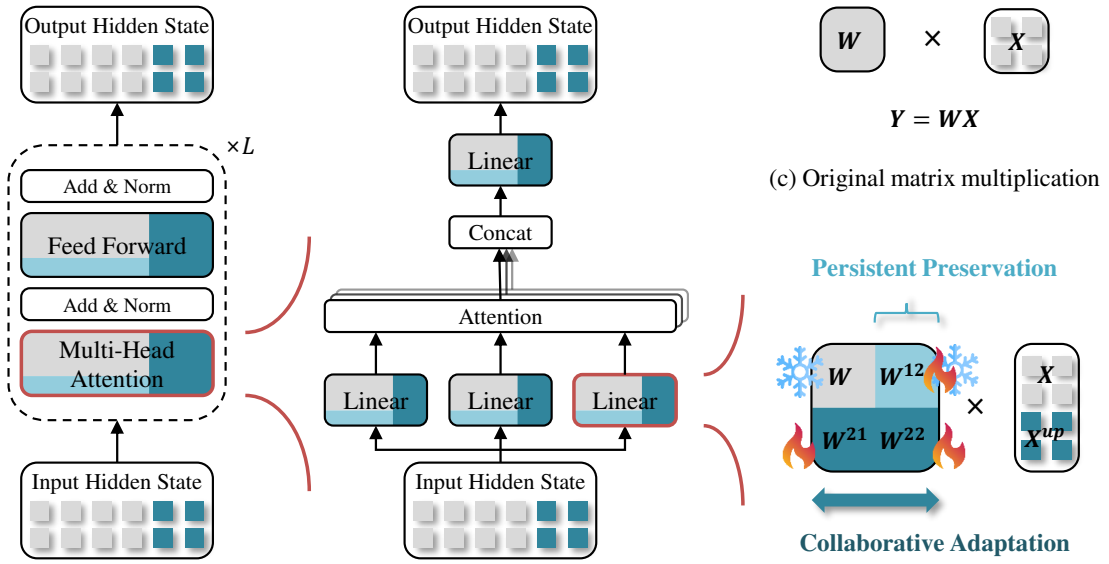
3. *Width-Upscaled Learning Methods.* We introduce SCALE-Preserve, SCALE-Adapt, and SCALE-Route; SCALE-Route performs token-level routing between preservation and adaptation paths and admits a tighter convergence bound than standard CL (Section 4).

4. *Empirical Validation.* Across continual biography learning and Korean CPT, SCALE reduces English-side forgetting while maintaining competitive Korean gains; SCALE-Route achieves the best stability–plasticity trade-off among evaluated baselines (FFT, LoRA, Freeze, LLaMA Pro) (Section 5).

2 Related Work

Continual Learning Continual learning (CL) adapts models to sequential tasks while mitigating catastrophic forgetting. Representative approaches include regularization (Kirkpatrick et al., 2017; Zenke et al., 2017; Aljundi et al., 2018), replay (Rolnick et al., 2019; Shin et al., 2017; Sun et al., 2019), and parameter isolation (Rusu et al., 2016; Li and Liang, 2021; Hu et al., 2022; Zhang et al., 2023). With the rise of LLMs, CL research has shifted toward preserving extensive pre-trained knowledge while incorporating new linguistic or domain-specific knowledge. Yet methods developed for smaller models or narrow tasks often do not transfer cleanly to LLMs due to computational and privacy constraints. Recent studies thus highlight parameter-efficient fine-tuning,(PEFT) and architectural extensions that allocate additional capacity for adaptation, enabling LLMs to retain general knowledge and improve in multilingual or specialized domains.

Upscaled Learning Model upscaling expands a pre-trained LLM to increase capacity while retaining performance. Depth upscaling adds layers (e.g., SOLAR (Kim et al., 2023), LLaMA Pro (Wu et al., 2024)) via duplication/interleaving followed by continual pre-training. Width upscaling increases hidden dimension or attention heads, often using function-preserving transformations (Chen et al., 2015, 2021). Several methods initialize larger models from smaller ones to reduce training overhead, spanning width upscaling (Shen et al., 2022; Yao et al., 2023; Samragh et al., 2024) and related depth-upscaling schemes. From a continual learning perspective, ELLE (Qin et al., 2022) and LOIRE (Han et al., 2025) use function-preserving depth and width upscaling to reduce catastrophic forgetting while incorporating new domain knowledge.



(a) Decoder layer with **Width Upscaling**

(b) Module example: MHA (same upscaling in FFN)

(c) Original matrix multiplication

(d) Upscaled matrix multiplication

Figure 3: **Overview of SCALE architecture.**

3 Proposed Architecture: SCALE

In this section, we introduce the width upscaling architecture SCALE and support its design with empirical evidence and theoretical analysis. We first give an overview in Section 3.1, then present two principles: ① *Persistent Preservation* and ② *Collaborative Adaptation*. Under *Persistent Preservation* (Section 3.2), SCALE zero-initializes and freezes $\boldsymbol{W}^{12}$ so that the original function remains intact throughout training. Under *Collaborative Adaptation* (Section 3.3), SCALE selectively trains expansion blocks to acquire new knowledge while minimizing interference with prior knowledge.

3.1 Overview of SCALE Architecture

Figure 3 shows an overview of the proposed width upscaling SCALE architecture. For decoder layers in Figure 3(a), we upscale the width of the input hidden state as well as the dimensions of the Multi-Head Attention (MHA) and Feed Forward Network (FFN), thus upscaling the width of the output hidden state. As illustrated in Figure 3(d), all matrix multiplications $\boldsymbol{W}\boldsymbol{X}$ in the MHA and FFN are expanded to their upscaled ones, formulated as:

$$\begin{bmatrix} \boldsymbol{W} & \boldsymbol{W}^{12} \\ \boldsymbol{W}^{21} & \boldsymbol{W}^{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{X} \\ \boldsymbol{X}^{up} \end{bmatrix} \tag{1}$$

where $\boldsymbol{W}^{12}$, $\boldsymbol{W}^{21}$, and $\boldsymbol{W}^{22}$ denote the upscaled weight matrix blocks and $\boldsymbol{X}^{up}$ denotes the upscaled part of the input. In particular, for MHA, this upscaling involves increasing the number of attention heads while keeping the head dimension fixed. In other words, with $\boldsymbol{W}$ being the query, key, and value projection matrices, the upscaled outputs

$$\boldsymbol{W}^{21}\boldsymbol{X} + \boldsymbol{W}^{22}\boldsymbol{X}^{up}$$

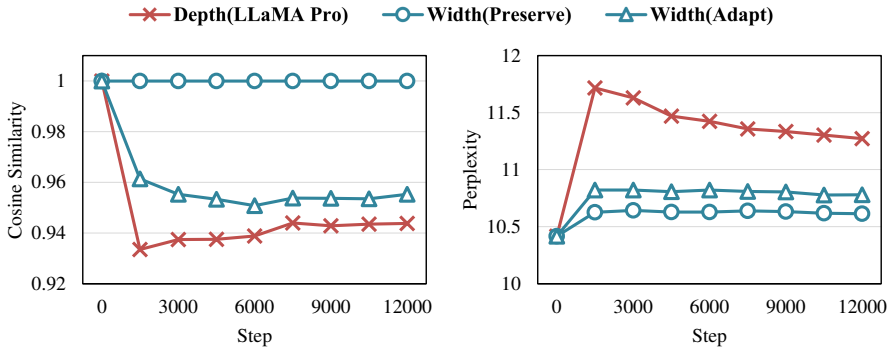
produce query/key/value representations for the new heads. Consequently, SCALE requires MHA weight matrices to be expanded so that the number of rows increases by an integer multiple of the head dimension. For embedding and output projection matrices, we upscale them as:

$$\boldsymbol{W}_{in} \mapsto \begin{bmatrix} \boldsymbol{W}_{in} \\ \boldsymbol{W}_{in}^{up} \end{bmatrix}, \quad \boldsymbol{W}_{out} \mapsto \begin{bmatrix} \boldsymbol{W}_{out} & \boldsymbol{W}_{out}^{up} \end{bmatrix}.$$

3.2 Design Principle 1: *Persistent Preservation*

Persistent Preservation prevents forgetting by keeping the original function $\boldsymbol{W}\boldsymbol{X}$ unchanged throughout training via a blockwise initialization-and-freeze scheme. The key requirement is to zero-initialize and freeze $\boldsymbol{W}^{12}$. We further initialize $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$ to mimic $\boldsymbol{W}$ and $\boldsymbol{W}^{12}$, respectively, to start from a well-conditioned expansion.

Initialization and Freeze of $\boldsymbol{W}^{12}$ To isolate function preservation, we compare depth upscaling (LLaMA Pro) and width upscaling (SCALE) by tracking how representation forgetting translates



(a) Representation forgetting.

(b) Forgetting perplexity.

Figure 4: **Comparison of depth and width upscaling from the perspective of how (a) representation forgetting causes (b) forgetting perplexity of pre-trained English knowledge across steps for new domain adaptation.**

into forgetting perplexity on pre-trained English knowledge[1] (Figure 4). Width(Preserve) zero-initializes and freezes $\boldsymbol{W}^{12}$ in all layers, while the remaining upscaled weights are randomly initialized and trainable. Width(Adapt) is identical, except $\boldsymbol{W}^{12}$ is trainable in all layers, making it non-preserving.

Figure 4a measures representation forgetting via cosine similarity between the last-layer outputs at step 0 and later steps (we compare only the original hidden-state part since depth upscaling does not widen it). As designed, Width(Preserve) never forgets the original function $\boldsymbol{W}\boldsymbol{X}$, whereas Depth(LLaMA Pro) drops sharply around step 1000 and struggles to recover. This gap reflects how much of the original computation is preserved: with zero-initialized frozen $\boldsymbol{W}^{12}$, width upscaling keeps $\boldsymbol{W}\boldsymbol{X}$ intact, while depth upscaling perturbs it immediately due to the inserted trainable layers. In contrast, Width(Adapt) forgets gradually and yields slightly higher forgetting perplexity than Width(Preserve), yet remains far below Depth(LLaMA Pro) (Figure 4b).

Theorem 3.1 formalizes this behavior: setting $\boldsymbol{W}^{12} = \mathbf{0}$ in every layer is necessary to preserve the original function, which is achieved by zero-initialization. Moreover, the contrast between Width(Preserve) and Width(Adapt) indicates that *freezing* $\boldsymbol{W}^{12}$, in addition to zero-initialization, is required for *Persistent Preservation*.

Theorem 3.1. *Width-upscaled network with* $\boldsymbol{W}_\ell^{12}$ *set to* $\mathbf{0}$ *preserves the original function for all layers* $1 \le \ell \le L$.

Proof. We defer the proof to Appendix B. □

[1]In this section, we perform CPT on FineWeb2 Korean data subset with Llama-3.2-1B as the base model of SCALE.

Initialization of $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$ Figure 5 compares initialization pairs for $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$: **0** (zero), **RND** (random; (He et al., 2015)), and **SVD** (dimension-reduced SVD of $\boldsymbol{W}$, as in LESA (Yang et al., 2025)). All pairs show near-perfect preservation on English (Figure 5a), confirming that $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$ do not affect function preservation. In contrast, Korean adaptation follows a clear ordering (Figure 5b): **0,RND** < **RND,0** < **0,SVD** < **SVD,0**. We therefore adopt **SVD,0** as the default for $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$, consistent with Corollary 3.2.

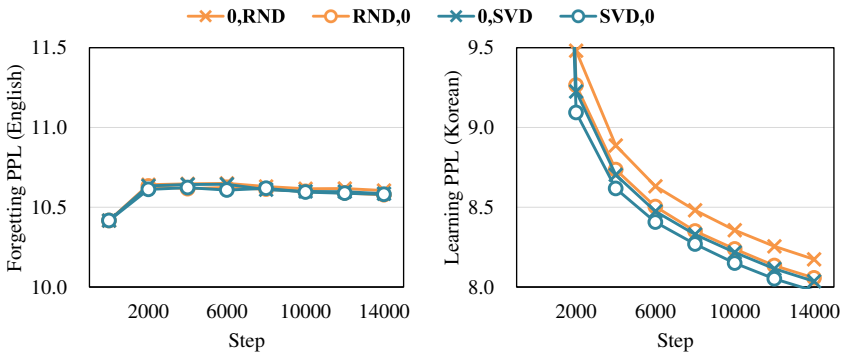
Corollary 3.2. $\boldsymbol{W}_\ell^{21}$ *and* $\boldsymbol{W}_\ell^{22}$ *can be initialized to other than* $\mathbf{0}$ *for new task adaptation without disrupting the original function because* $\boldsymbol{W}_\ell^{21}$ *and* $\boldsymbol{W}_\ell^{22}$ *are irrelevant to the function preservation.*

3.3 Design Principle 2: *Collaborative Adaptation*

Collaborative Adaptation enables SCALE to acquire new-domain knowledge by training only selected upscaled blocks. We explore which layers and modules to train for effective collaboration.

Collaborative Layers Freezing $\boldsymbol{W}^{12}$ (for *Persistent Preservation*) cleanly decouples $\boldsymbol{X}$ and $\boldsymbol{X}^{up}$, but this also limits learning capacity. To characterize controlled collaboration, we keep $\boldsymbol{W}^{12}$ zero-initialized and allow $\boldsymbol{W}^{12}$ to be trainable only in upper layers, while preserving $\boldsymbol{W}^{12}$ in lower layers. Let L_{fp} denote the number of lower layers whose $\boldsymbol{W}^{12}$ blocks remain function-preserving (frozen). Figure 6 shows a clear forgetting–learning trade-off as L_{fp} varies, with forgetting increasing exponentially as L_{fp} decreases. Proposition 3.3 and Corollary 3.4 support this trend.

Proposition 3.3. *The accumulated output shift of*



(a) Forgetting perplexity on test English data. (b) Learning perplexity on test Korean data.

Figure 5: **Comparison of initialization pairs for W^{21} and W^{22}. Each method pair is separated by a comma.**

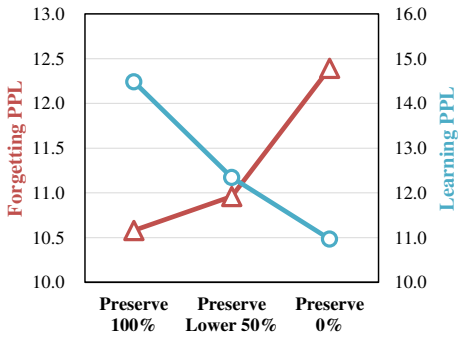


Figure 6: **Forgetting-learning PPL trade-off according to the amount of preserving lower layers L_{fp}.**

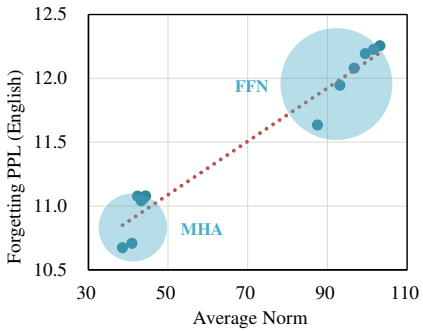


Figure 7: **Near-linear scalability of forgetting perplexity to upscaled weight norm.**

the width-upscaled residual network with function-preserving lower layers $1 \leq \ell \leq L_{fp}$ and non-preserving upper layers $L_{fp} < \ell \leq L$ is bounded by Eq. (18).

$$\left\| \tilde{\boldsymbol{X}}_L^{UP} - \boldsymbol{X}_L^{UP} \right\| \leq (L - L_{fp})\epsilon(1+\delta_{np})^{L-1}\left(\frac{1+\delta_{fp}}{1+\delta_{np}}\right)^{L_{fp}} \left\| \boldsymbol{X}_0^{UP} \right\| \quad (2)$$

where $\tilde{\boldsymbol{X}}_L^{UP}$ denotes updated output of width-upscaled residual network as defined by Definition C.3 and δ_{fp} and δ_{np} denotes upper bound of norm of upscaled weight matrix for function-preserving lower layers and non-preserving upper layers, respectively, under Assumption C.5.

Proof. We defer the proof to Appendix C. □

Corollary 3.4. *Forgetting increases exponentially with decreasing L_{fp}, the number of function preserving $\boldsymbol{W}_\ell^{12}$ blocks in lower layers $1 \leq \ell \leq L_{fp}$.*

Collaborative Modules Beyond layer choice, we compare where to collaborate within a layer. With function-preserving lower half layers ($L_{fp} = L/2$), we train for one epoch while making $\boldsymbol{W}^{12}$ trainable only in the MHA or the FFN, enabling module-specific collaboration. Figure 7 shows that forgetting perplexity scales near-linearly with the average norm of the upscaled weights in both cases. However, FFN collaboration incurs much larger forgetting (its intermediate dimension is large), and the converged perplexity remains far above the baseline (e.g., ∼10.4 for LLaMA-3.2-1B), making it an undesirable choice. In contrast, MHA-only collaboration converges with a smaller error bound and only a slight perplexity increase, suggesting MHA as the preferred collaborative module.

4 Proposed Learning Methods

The two principles above naturally yield complementary learning methods for upscaled continual learning: preservation-first and adaptation-first. Since neither alone is universally optimal, we also combine their strengths to mitigate the stability–plasticity trade-off (Figure 6). Concretely, we propose three learning methods: ① SCALE-Preserve, ② SCALE-Adapt, and ③ SCALE-Route.

- SCALE-Preserve is preservation-first: for all ℓ, $\boldsymbol{W}_\ell^{12}$ is initialized to $\mathbf{0}$ and kept frozen, yielding Eq. (3),

$$\boldsymbol{Z}_{preserve}^{UP} \triangleq \boldsymbol{Z}_{preserve} + \boldsymbol{Z}_{preserve}^{up} \tag{3}$$

where $\boldsymbol{Z} = \boldsymbol{W}_{out}\boldsymbol{X}_L$ and $\boldsymbol{Z}^{up} = \boldsymbol{W}_{out}^{up}\boldsymbol{X}_L^{up}$ denote output logits for original and upscaled part, respectively.

- SCALE-Adapt is adaptation-first: for all ℓ, $\boldsymbol{W}_\ell^{12}$ is initialized to $\mathbf{0}$ but made trainable, yielding Eq. (4).

$$\boldsymbol{Z}_{adapt}^{UP} \triangleq \boldsymbol{Z}_{adapt} + \boldsymbol{Z}_{adapt}^{up} \tag{4}$$

- SCALE-Route combines both behaviors via token-level routing, aiming to maximize preservation and adaptation simultaneously and thus mitigate the trade-off in Figure 6. Since the key difference between SCALE-Preserve and SCALE-Adapt is whether $\boldsymbol{W}^{12}$ is trainable, SCALE-Route selects computation paths by comparing their logits. Specifically, it uses cosine similarity as a router with threshold τ (Eq. (5)); we fix $\tau = 0.5$ as a robust default across settings. If $\boldsymbol{Z}_{preserve}$ and $\boldsymbol{Z}_{adapt}$ agree (high cosine similarity), we route to SCALE-Preserve; otherwise we route to SCALE-Adapt to exploit adaptation opportunities. We approximate $\boldsymbol{Z}_{preserve}$ by $(\boldsymbol{Z}_{preserve}+\boldsymbol{Z}_{adapt})/2$ due to better trainability of $\boldsymbol{Z}_{adapt}$. Finally, because SCALE-Adapt can reproduce SCALE-Preserve whenever $\boldsymbol{W}^{12}$ effectively evaluates to $\mathbf{0}$, both logits can be obtained within a single forward pass with only slight extra computation.

$$\boldsymbol{Z}_{route}^{UP} \triangleq \begin{cases} \dfrac{\boldsymbol{Z}_{preserve} + \boldsymbol{Z}_{adapt}}{2} + \boldsymbol{Z}_{preserve}^{up} \\ \qquad \text{if } \cos(\boldsymbol{Z}_{preserve}, \boldsymbol{Z}_{adapt}) > \tau \\ \boldsymbol{Z}_{adapt}^{UP} \qquad \text{otherwise} \end{cases} \tag{5}$$

Theorem 4.1 provides an analysis that supports the superiority of SCALE-Route.

Theorem 4.1. Routing-based Continual Learning *with logit routing admits a tighter convergence bound than standard* Continual Learning *by gating interference updates.*

Proof. The proof is deferred to Appendix D. □

5 Experiments

5.1 Experimental Setup

The Biography Dataset We reproduce the controlled experiment in (Zheng et al., 2025) to compare forgetting across continual learning methods, focusing on SCALE-Route. The *Biography Dataset*[2] contains 200,000 synthetic individuals with a name and six attributes (birthday, birth city, university, major, company name, company city), split into pre-training and fine-tuning data. Following (Zheng et al., 2025), we use three stages: pre-train on the first 100,000 individuals, fine-tune QA on the first 50,000 (Task 0), then apply an upscaling method and fine-tune QA on 20,000 unseen individuals (Task 1). During Task 1, we monitor Task 0 degradation using *hard first-token accuracy* (whether the top-predicted first token is correct).

We utilize the Pythia-160M (Biderman et al., 2023) architecture as our backbone model. For SCALE-Route, we upscale both the hidden dimension and the FeedForward dimension by 128, and train $\boldsymbol{W}^{12}$ only for the last 12th layer. In order to match the number of trainable parameters in SCALE-Route, LLaMA Pro expands the number of layers from 12 to 16. We note that LLaMA Pro relies on the LLaMA architecture, where the output weight matrices are zero-initialized in the expanded block to preserve the output from the initial model. In contrast, since Pythia adopts a GPT-NeoX (Andonian et al., 2023) architecture, a different set of the output weight matrices should be zero-initialized. We use the same hyperparameter settings as in (Zheng et al., 2025), except that, during Task 1 learning, SCALE-Route and LLaMA Pro are trained with an increased learning rate of 5×10^{-5}, which is ten times larger than the original setting. This modification is necessary because both methods freeze a substantial portion of the parameters, and therefore an increased learning rate is required to ensure performance on Task 1. The experiments are executed on an NVIDIA H100 80GB GPU.

Continual Pre-training In order to investigate forgetting phenomena, we constrain the training data to the Korean subset of FineWeb2 (Penedo et al., 2025), a 60-billion-token Korean web data filtered from Common Crawl. We deliberately exclude data from other domains, since the presence of English corpus can induce data-replay effects,

[2] https://github.com/zzz47zzz/spurious-forgetting

which in turn hinder a precise comparison among upscaling methods.

For each method, we initialize our base model with LLaMA3.2-1B and perform continual pre-training on the Korean dataset for one epoch, using a batch size of 512, a sequence length of 8192, and a linear learning rate schedule with a warm-up ratio of 6%. Our SCALE methods upscale both the hidden dimension and the FeedForward dimension by 256 and 1024, respectively. We note that the base model is configured with a head dimension of 64 and uses the Grouped-Query Attention(GQA) with 4 KV projections, and therefore upscaling the hidden dimension by 256 represents a minimal upscaling. For SCALE-Adapt and SCALE-Route, we choose $L_{fp} = 3$. We also manually configure the hyperparameters of LLaMA Pro and LoRA to match the number of trainable parameters in SCALE methods. We expand the number of layers from 16 to 20 for LLaMA Pro, and use a rank of 256 and target all weight matrices in the MHA and FFN for LoRA. Freeze (Zheng et al., 2025) refers to freezing all components in the bottom three layers of the model, including the input embedding layer.

Finally, we set different learning rates due to the trade-off between learning and forgetting: 1×10^{-5} for FFT, LoRA and Freeze, 2×10^{-4} for LLaMA Pro, and 1×10^{-3} for SCALE. We adjust the learning rate individually for each experiment to achieve comparable learning performance, allowing us to fairly compare model's forgetting under similar learning conditions. Compared to FFT, upscaling methods employ higher learning rates because only newly added parameters are updated, requiring larger updates to achieve sufficient adaptation. In particular, SCALE requires an even larger learning rate than LLaMA Pro, as its strong function-preserving property suppresses early representation changes. All experiments are executed on 8 NVIDIA H100 80GB GPUs.

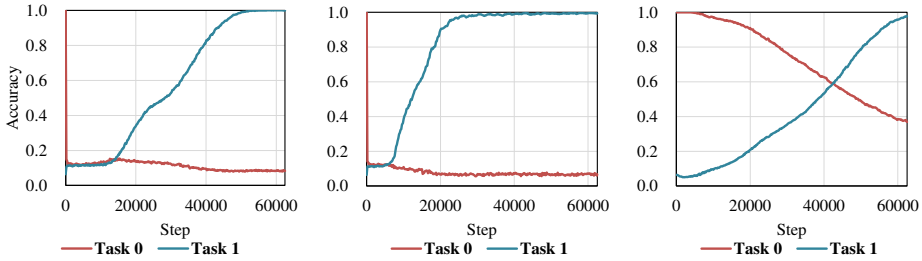
5.2 Results and Analysis

The Biography Dataset Results For the Biography Dataset experiment, we present the accuracy for Task 0 and Task 1 during Task 1 learning in Figure 8. We observe that for FFT and LLaMA Pro, the accuracy for Task 0 sharply drops to approximately 15% only after 200 steps, whereas for SCALE-Route it remains at 100% throughout the first 4000 steps of Task 1 learning. Furthermore, for SCALE-Route the final accuracy for Task 0 is 36.9% which is much higher compared to FFT and LLaMA Pro, highlighting its robustness against forgetting.

Another notable observation is that in Figure 8c, the accuracy curves of SCALE-Route for Task 0 and Task 1 gradually decrease and increase, respectively, showing smooth transitions without drops or spikes during Task 1 learning. This indicates that by varying the number of collaborative layers, the trade-off between forgetting and learning can be controlled in our architecture-based method, rather than a data replay-based method, aligning it with its learning objective.

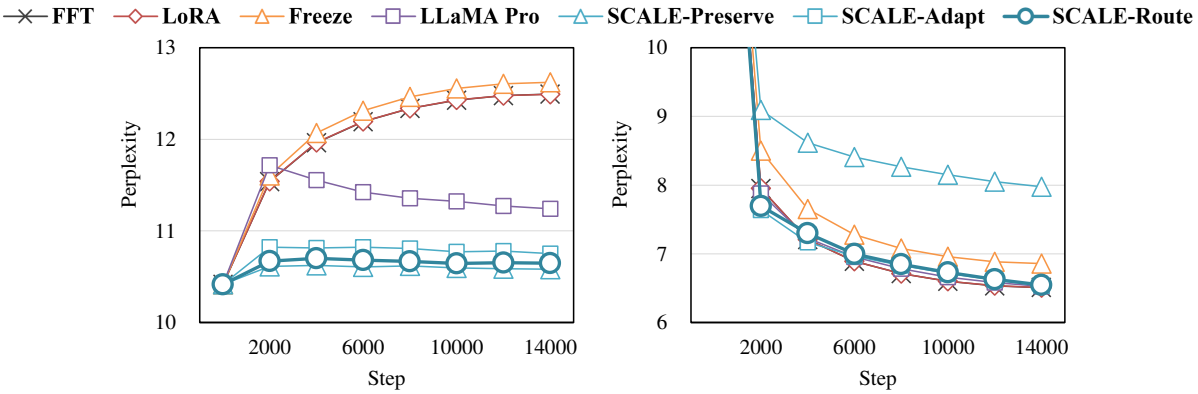
Continual Pre-Training Results We first analyze the perplexity on 30K samples of FineWeb-Edu and on the test split of the Korean subset of FineWeb2. As shown in Figure 9b, the perplexity on the Korean test data is almost identical across all methods except SCALE-Preserve, which is consistent with our intended design. In contrast, Figure 9a shows that our SCALE methods achieve lower perplexity on the English test data than the other methods. Compared to LLaMA Pro, SCALE shows a smaller increase in perplexity on the English test data in the early stage of training, and therefore it can be regarded as a more stable approach for Continual Pre-Training. We also observe that the perplexity gap between SCALE-Preserve and SCALE-Adapt arises more from learning than from forgetting, which suggests that training $\boldsymbol{W}^{12}$ more strongly affects learning than forgetting.

Furthermore, we evaluate our SCALE methods on the English and Korean benchmarks. Evaluations are conducted using Eleuther AI Language Model Evaluation Harness and in a zero shot setting. Due to the lack of instruction-following capability of pre-trained models, we select only a very limited set of Korean benchmarks, KoBEST (Jang et al., 2022), for evaluation. The results are presented in Table 1. We find that all SCALE methods preserve the original capabilities on English benchmarks, outperforming other methods. Although SCALE-Route obtains some improvement on the Korean benchmarks, it achieves only marginal improvement compared to FFT and LoRA. However, as SCALE-Adapt and SCALE-Route outperform SCALE-Preserve on the Korean benchmarks, expanding the training scope of $\boldsymbol{W}^{12}$ could yield further improvements while preserving the original capabilities.



(a) FFT. (b) LLaMA Pro. (c) SCALE-Route.

Figure 8: **Continual adaptation to biography Task 1 while preserving Task 0.**



(a) Forgetting perplexity on test English data. (b) Learning perplexity on test Korean data.

Figure 9: **Performance comparison of upscaling methods trained on FineWeb2 Korean subset for one epoch.**

Model	English						Korean			
	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	Avg.	KB BoolQ	KB COPA	KB HellaSwag	Avg.
Llama-3.2-1B	36.60	63.66	36.76	37.73	60.93	47.14	49.86	53.00	50.60	51.15
FFT	32.68	57.32	26.01	38.82	59.12	42.79	52.14	65.30	54.20	57.21
LoRA	32.85	57.17	25.76	38.99	58.17	42.59	51.85	64.90	55.00	57.25
LLaMA Pro	34.22	60.61	34.42	36.20	57.30	44.55	51.42	63.20	52.80	**55.81**
Freeze	31.23	56.59	26.43	38.67	59.27	42.44	50.50	60.10	53.40	54.67
SCALE-Preserve	36.52	62.75	33.79	37.89	59.51	**46.09**	52.42	57.60	49.40	53.14
SCALE-Adapt	34.81	61.85	35.22	38.25	60.62	**46.15**	50.36	63.10	51.80	**55.09**
SCALE-Route	35.84	61.72	36.31	37.50	61.09	**46.49**	51.50	63.80	51.20	**55.50**

Table 1: **Performance comparison of upscaling methods trained on FineWeb2 Korean subset for one epoch.**

6 Conclusion

This work presented SCALE, an architectural expansion recipe for stable continual pre-training without altering a model's computation graph. By widening linear submodules while freezing original parameters, SCALE adds capacity without changing the learned function; *Persistent Preservation* and *Collaborative Adaptation* define a stability–plasticity trade-off, with collaboration best placed in upper-layer attention when stability matters.

We instantiate these principles with three variants: SCALE-Preserve for maximal stability, SCALE-Adapt for maximal plasticity via broad collaboration, and SCALE-Route, which routes tokens between preservation and adaptation paths via a similarity criterion with little overhead.

Width upscaling outperforms depth expansion on the biography task, yielding higher retention with competitive adaptation. In continual pre-training on Korean web data, SCALE variants reduce forgetting on English evaluations while matching Korean learning, and SCALE-Route achieves the best stability–plasticity balance. Our theory explains preservation under initialization/freezing and why routing improves convergence.

Future work includes larger backbones and longer horizons, adaptive collaboration and routing, and integration with parameter-efficient tuning, retrieval, and broader multilingual/domain CPT, positioning width-upscaled continual learning beyond brute-force scaling.

7 Limitations

Our study primarily evaluates SCALE in controlled continual settings (e.g., a synthetic biography stream and a Korean-domain continual pretraining setup), so results may not fully generalize to broader task mixtures or instruction-tuned regimes; nonetheless, the method is simple to apply (freezing the base and training only expansion modules) and consistently improves stability–plasticity trade-offs within the tested scope. Potential risks include (i) added parameters and routing logic increasing compute/memory and operational complexity, (ii) sensitivity to design choices such as expansion size and routing thresholds that may require modest tuning, and (iii) unmeasured shifts in safety-relevant behaviors (e.g., bias/toxicity, privacy leakage, or unintended capability changes) that should be assessed with standard alignment and red-teaming evaluations prior to deployment.

References

Kwangjun Ahn, Xiang Cheng, Minhak Song, Chulhee Yun, Ali Jadbabaie, and Suvrit Sra. 2023. Linear attention is (maybe) all you need (to understand transformer optimization). *arXiv preprint arXiv:2310.01082*.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pages 139–154.

Alex Andonian, Quentin Anthony, Stella Biderman, Sid Black, Preetham Gali, Leo Gao, Eric Hallahan, Josh Levy-Kramer, Connor Leahy, Lucas Nestler, Kip Parker, Michael Pieler, Jason Phang, Shivanshu Purohit, Hailey Schoelkopf, Dashiell Stander, Tri Songz, Curt Tigges, Benjamin Thérien, and 2 others. 2023. GPT-NeoX: Large Scale Autoregressive Language Modeling in PyTorch. `https://www.github.com/eleutherai/gpt-neox`.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, and 1 others. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, and 1 others. 2022. Gpt-neox-20b: An open-source autoregressive language model. *arXiv preprint arXiv:2204.06745*.

Enric Boix-Adsera, Etai Littwin, Emmanuel Abbe, Samy Bengio, and Joshua Susskind. 2023. Transformers learn through gradual rank increase. *Advances in Neural Information Processing Systems*, 36:24519–24551.

Cheng Chen, Yichun Yin, Lifeng Shang, Xin Jiang, Yujia Qin, Fengyu Wang, Zhi Wang, Xiao Chen, Zhiyuan Liu, and Qun Liu. 2021. bert2bert: Towards reusable pretrained language models. *arXiv preprint arXiv:2110.07143*.

Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. 2015. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*.

Manas Deb and Tokunbo Ogunfunmi. 2025. Information-theoretical analysis of a transformer-based generative ai model. *Entropy*, 27(6):589.

Wenyu Du, Tongxu Luo, Zihan Qiu, Zeyu Huang, Yikang Shen, Reynold Cheng, Yike Guo, and Jie Fu. 2024. Stacking your transformers: A closer look at model growth for efficient llm pre-training. *arXiv preprint arXiv:2405.15319*.

Xue Han, Yitong Wang, Junlan Feng, Qian Hu, Chao Deng, and 1 others. 2025. Loire: Lifelong learning on incremental data via pre-trained language model growth efficiently. In *The Thirteenth International Conference on Learning Representations*.

Bobby He, James Martens, Guodong Zhang, Aleksandar Botev, Andrew Brock, Samuel L Smith, and Yee Whye Teh. 2023. Deep transformers without shortcuts: Modifying self-attention for faithful signal propagation. *arXiv preprint arXiv:2302.10322*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Myeongjun Jang, Dohyung Kim, Deuk Sin Kwon, and Eric Davis. 2022. Kobest: Korean balanced evaluation of significant tasks. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3697–3708.

Dahyun Kim, Chanjun Park, Sanghoon Kim, Wonsung Lee, Wonho Song, Yunsu Kim, Hyeonwoo Kim, Yungi Kim, Hyeonju Lee, Jihoo Kim, and 1 others. 2023. Solar 10.7 b: Scaling large language models with simple yet effective depth up-scaling. *arXiv preprint arXiv:2312.15166*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, and 1 others. 2017.

Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Yann LeCun. 2025. Meta's Yann LeCun: Scaling AI Won't Make It Smarter. "https://www.businessinsider.com/meta-yann-lecun-scaling-ai-wont-make-it-smarter-2025-4".

Hongkang Li, Meng Wang, Sijia Liu, and Pin-Yu Chen. 2023. A theoretical understanding of shallow vision transformers: Learning, generalization, and sample complexity. *arXiv preprint arXiv:2302.06015*.

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

Toan Q Nguyen and Julian Salazar. 2019. Transformers without tears: Improving the normalization of self-attention. *arXiv preprint arXiv:1910.05895*.

Guilherme Penedo, Hynek Kydlívcek, Vinko Sabolvcec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. 2025. Fineweb2: One pipeline to scale them all – adapting pre-training data processing to every language. *Preprint*, arXiv:2506.20920.

Yujia Qin, Jiajie Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2022. Elle: Efficient lifelong pre-training for emerging data. *arXiv preprint arXiv:2203.06311*.

Zhen Qin, Jinxin Zhou, and Zhihui Zhu. 2025. On the convergence of gradient descent on learning transformers with residual connections. *arXiv preprint arXiv:2506.05249*.

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

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Mohammad Samragh, Iman Mirzadeh, Keivan Alizadeh Vahid, Fartash Faghri, Minsik Cho, Moin Nabi, Devang Naik, and Mehrdad Farajtabar. 2024. Scaling smart: Accelerating large language model pre-training with small model initialization. *arXiv preprint arXiv:2409.12903*.

Sheng Shen, Pete Walsh, Kurt Keutzer, Jesse Dodge, Matthew Peters, and Iz Beltagy. 2022. Staged training for transformer language models. In *International Conference on Machine Learning*, pages 19893–19908. PMLR.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2019. Lamol: Language modeling for lifelong language learning. *arXiv preprint arXiv:1909.03329*.

Ilya Sutskever. 2024. OpenAI and others seek new path to smarter AI as current methods hit limitations. "https://www.reuters.com/technology/artificial-intelligence/openai-rivals-seek-new-path-smarter-ai-current-methods-hit-limitations-2024-11-11".

Ben Wang. 2021. Mesh-transformer-jax: model-parallel implementation of transformer language model with jax.

Peihao Wang, Rameswar Panda, Lucas Torroba Hennigen, Philip Greengard, Leonid Karlinsky, Rogerio Feris, David Daniel Cox, Zhangyang Wang, and Yoon Kim. 2023. Learning to grow pretrained models for efficient transformer training. *arXiv preprint arXiv:2303.00980*.

Chengyue Wu, Yukang Gan, Yixiao Ge, Zeyu Lu, Jiahao Wang, Ye Feng, Ying Shan, and Ping Luo. 2024. LLaMA Pro: Progressive LLaMA with Block Expansion. *arXiv preprint arXiv:2401.02415*.

Yifei Yang, Zouying Cao, Xinbei Ma, Yao Yao, Libo Qin, Zhi Chen, and Hai Zhao. 2025. Lesa: Learnable llm layer scaling-up. *arXiv preprint arXiv:2502.13794*.

Yiqun Yao, Zheng Zhang, Jing Li, and Yequan Wang. 2023. Masked structural growth for 2x faster language model pre-training. *arXiv preprint arXiv:2305.02869*.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR.

Shuangfei Zhai, Walter Talbott, Nitish Srivastava, Chen Huang, Hanlin Goh, Ruixiang Zhang, and Josh Susskind. 2021. An attention free transformer. *arXiv preprint arXiv:2105.14103*.

Renrui Zhang, Jiaming Han, Chris Liu, Peng Gao, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, and Yu Qiao. 2023. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*.

Junhao Zheng, Xidi Cai, Shengjie Qiu, and Qianli Ma. 2025. Spurious Forgetting in Continual Learning of Language Models. *arXiv preprint arXiv:2501.13453*.

A Width Upscaling

We consider a simplified decoder model that retains residual connections while omitting modules such as MHA (multi-head attention) or FFN (feed-forward networks). This abstraction enables us to concentrate on the dynamics of residual layers, particularly their roles in balancing **preservation** and **adaptation** of representations. Such model reduction is consistent with a broader line of theoretical work that introduces simplifications to isolate core mechanisms, as highlighted in Remark A.1

Remark A.1. Theoretical analysis of the full Transformer architecture is notoriously challenging due to the intricate interaction of its components. Recent studies have therefore adopted simplified settings, such as single-layer or shallow Transformers (Li et al., 2023), restricted weight structures (Boix-Adsera et al., 2023), and attention-free or linearized variants (Ahn et al., 2023; Zhai et al., 2021), to obtain tractable insights. Our focus on a residual-only decoder follows this tradition, motivated by growing evidence that residual connections form the backbone of stable signal propagation and representation transport in Transformers (He et al., 2023; Qin et al., 2025; Deb and Ogunfunmi, 2025)

Definition A.2. (Residual Network)

$$\boldsymbol{X}_\ell \triangleq (\boldsymbol{W}_\ell + \boldsymbol{I})\boldsymbol{X}_{\ell-1} \quad (1 \le \ell \le L) \tag{6}$$

where each ℓ-th layer outputs hidden states $\boldsymbol{X}_\ell \in \mathbb{R}^d$ and has a weight matrix $\boldsymbol{W}_\ell \in \mathbb{R}^{d\times d}$, $\boldsymbol{X}_0$ denotes embedding input, and $\boldsymbol{I} \in \mathbb{R}^{d\times d}$ denotes the identity matrix.

Next, we upscale the width of the residual network, as defined in Definition A.3. It follows general formulations in upscaled learning literature (Du et al., 2024; Shen et al., 2022; Samragh et al., 2024).

Definition A.3. (Residual Network with Width Upscaling)

$$\boldsymbol{X}_\ell^{UP} \triangleq (\boldsymbol{W}_\ell^{UP} + \boldsymbol{I}^{UP})\boldsymbol{X}_{\ell-1}^{UP} \quad (1 \le \ell \le L) \tag{7}$$

where $\boldsymbol{X}_\ell\,(0 \le \ell \le L)$ is upscaled on dimension of width to $\boldsymbol{X}_\ell^{UP} \triangleq \begin{bmatrix} \boldsymbol{X}_\ell \\ \boldsymbol{X}_\ell^{up} \end{bmatrix} \in \mathbb{R}^{(d+d_{up})}$, $\boldsymbol{W}_\ell\,(1 \le \ell \le L)$ is upscaled accordingly to $\boldsymbol{W}_\ell^{UP} \triangleq \begin{bmatrix} \boldsymbol{W}_\ell & \boldsymbol{W}_\ell^{12} \\ \boldsymbol{W}_\ell^{21} & \boldsymbol{W}_\ell^{22} \end{bmatrix} \in \mathbb{R}^{(d+d_{up})\times(d+d_{up})}$, and $\boldsymbol{I}$ is also upscaled to $\boldsymbol{I}^{UP} \triangleq \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}^{up} \end{bmatrix} \in \mathbb{R}^{(d+d_{up})\times(d+d_{up})}$.

B Function-Preserving Width Upscaling

Function preservation serves as a foundational principle in model upscaling (Shen et al., 2022; Wang et al., 2023; Yao et al., 2023; Samragh et al., 2024; Qin et al., 2022; Han et al., 2025), aiming to ensure that the functional behavior of the model remains unchanged despite architectural modifications.

Formally, F denotes the original function (e.g., an end-to-end function or just a layer), taking $\boldsymbol{X}$ as the input (e.g., input tokens or input hidden states). By transforming or expanding $F(\boldsymbol{X})$, we can upscale $F : \boldsymbol{X} \to \boldsymbol{Y}$ to $\mathcal{F} : \boldsymbol{X} \times \boldsymbol{U} \to \boldsymbol{Y} \times \boldsymbol{V}$. Then, the objective of function preservation is to satisfy Eq. (8).

$$\forall \boldsymbol{X}, \pi_{\boldsymbol{Y}}(\mathcal{F}(\boldsymbol{X}, \boldsymbol{U})) = F(\boldsymbol{X}) \tag{8}$$

where $\pi_{\boldsymbol{Y}}$ denotes a projection function to $\boldsymbol{Y}$. That is, F is preserved in $\mathcal{F}$.

This formulation enables model upscaling by allowing new knowledge to be learned, while ensuring that the original knowledge remains unforgotten.

To satisfy function preservation in the width-upscaled residual network in Definition A.3, it is the simplest to set $\boldsymbol{W}_\ell^{12}$ to $\boldsymbol{0}$ for all layers $1 \le \ell \le L$, as defined in Definition B.1.

Definition B.1. (Residual Network with Function-Preserving Width Upscaling).

$$\boldsymbol{X}_\ell^{UP} \triangleq \left(\begin{bmatrix} \boldsymbol{W}_\ell & \boldsymbol{0} \\ \boldsymbol{W}_\ell^{21} & \boldsymbol{W}_\ell^{22} \end{bmatrix} + \boldsymbol{I}^{UP} \right) \boldsymbol{X}_{\ell-1}^{UP} \tag{9}$$

Theorem 3.1. *Width-upscaled network with $\boldsymbol{W}_\ell^{12}$ set to $\boldsymbol{0}$ preserves the original function $\boldsymbol{X}_\ell = (\boldsymbol{W}_\ell + \boldsymbol{I})\boldsymbol{X}_{\ell-1}$ for all layers $1 \le \ell \le L$.*

Proof. From Eq. (7), we can express the original function as $F(\boldsymbol{X}_{\ell-1}) = (\boldsymbol{W}_\ell + \boldsymbol{I})\boldsymbol{X}_{\ell-1}$ and its width-upscaled function as $\mathcal{F}\big(\boldsymbol{X}_{\ell-1}^{UP}\big) = \begin{bmatrix} \boldsymbol{W}_\ell + \boldsymbol{I} & \boldsymbol{W}_\ell^{12} \\ \boldsymbol{W}_\ell^{21} & \boldsymbol{W}_\ell^{22} + \boldsymbol{I}^{up} \end{bmatrix} \begin{bmatrix} \boldsymbol{X}_{\ell-1} \\ \boldsymbol{X}_{\ell-1}^{up} \end{bmatrix}$. By setting $\boldsymbol{W}_\ell^{12}$ of $\mathcal{F}\big(\boldsymbol{X}_{\ell-1}^{UP}\big)$ to $\boldsymbol{0}$, it can be easily shown by Eq. (10) that Eq. (9) satisfies function preservation because for all $\boldsymbol{X}_{\ell-1}^{UP}$, there exists a projection

function $\pi_{\boldsymbol{Y}}\left(\mathcal{F}\left(\boldsymbol{X}_{\ell-1}^{UP}\right)\right) = F(\boldsymbol{X}_{\ell-1})$.

$$\begin{aligned}\mathcal{F}\left(\boldsymbol{X}_{\ell-1}^{UP}\right) &= \begin{bmatrix}\boldsymbol{W}_\ell + \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{W}_\ell^{21} & \boldsymbol{W}_\ell^{22} + \boldsymbol{I}^{up}\end{bmatrix}\begin{bmatrix}\boldsymbol{X}_{\ell-1} \\ \boldsymbol{X}_{\ell-1}^{up}\end{bmatrix} \\ &= \begin{bmatrix}(\boldsymbol{W}_\ell + \boldsymbol{I})\boldsymbol{X}_{\ell-1} \\ \boldsymbol{W}_\ell^{21}\boldsymbol{X}_{\ell-1} + \left(\boldsymbol{W}_\ell^{22} + \boldsymbol{I}^{up}\right)\boldsymbol{X}_{\ell-1}^{up}\end{bmatrix} \\ &= \begin{bmatrix}F(\boldsymbol{X}_{\ell-1}) \\ \boldsymbol{W}_\ell^{21}\boldsymbol{X}_{\ell-1} + \left(\boldsymbol{W}_\ell^{22} + \boldsymbol{I}^{up}\right)\boldsymbol{X}_{\ell-1}^{up}\end{bmatrix}\end{aligned} \tag{10}$$

From the recursion of layers in Eq (10), function preservation of the end-to-end network is also satisfied. □

Corollary 3.2. *$\boldsymbol{W}_\ell^{21}$ and $\boldsymbol{W}_\ell^{22}$ can be initialized to other than $\boldsymbol{0}$ for new task adaptation without disrupting the original function because $\boldsymbol{W}_\ell^{21}$ and $\boldsymbol{W}_\ell^{22}$ values are irrelevant to the function preservation.*

Proof. Immediate from the proof of Theorem 3.1. □

C Forgetting Analysis

In this section, we analyze forgetting from the perspective of accumulated output shift. First of all, we make the following assumptions, as in many other relevant studies (Zheng et al., 2025; Wang, 2021; Nguyen and Salazar, 2019; Black et al., 2022).

Assumption C.1. (Small Weight Norm). For every layer ℓ, the norm of its upscaled weight matrix is bounded by a small constant $\delta > 0$, i.e., $\left\|\boldsymbol{W}_\ell^{UP}\right\| \leq \delta$.

Assumption C.2. (Small Gradient Norm). For every layer ℓ, the norm of its upscaled gradient matrix is bounded by a small constant $\epsilon > 0$, i.e., $\left\|\Delta\boldsymbol{W}_\ell^{UP}\right\| \leq \epsilon$.

Definition C.3. (Updated Output of Width-Upscaled Residual Network) By updating the upscaled weight matrix $\boldsymbol{W}_\ell^{UP}$ in Eq. (7) to $\tilde{\boldsymbol{W}}_\ell^{UP} \triangleq \boldsymbol{W}_\ell^{UP} + \Delta\boldsymbol{W}_\ell^{UP}$, the corresponding updated output is defined as Eq. (11).

$$\tilde{\boldsymbol{X}}_\ell^{UP} \triangleq \left(\boldsymbol{W}_\ell^{UP} + \Delta\boldsymbol{W}_\ell^{UP} + \boldsymbol{I}^{UP}\right)\tilde{\boldsymbol{X}}_{\ell-1}^{UP} \tag{11}$$

Based on Assumptions C.1 and C.2, we analyze the output shift bound of the width upscaling from Definition A.3 in Proposition C.4 and extend it to the function-preserving width upscaling from Definition B.1 in Proposition 3.3. Note that this is also an extension of *Proposition 4.9 from* Freeze (Zheng et al., 2025).

Proposition C.4. *The accumulated output shift for all layers $1 \leq \ell \leq L$ of the residual network with width upscaling is bounded by Eq. (12).*

$$\left\|\tilde{\boldsymbol{X}}_L^{UP} - \boldsymbol{X}_L^{UP}\right\| \leq L\epsilon(1+\delta)^{L-1}\left\|\boldsymbol{X}_0^{UP}\right\| \tag{12}$$

Proof. We begin by deriving accumulated output in Eq. (13) and accumulated output after a learning step in Eq. (14) from recursion of Eq. (7) and Eq. (11), respectively.

$$\boldsymbol{X}_L^{UP} = \prod_{\ell=1}^{L}\left(\boldsymbol{W}_\ell^{UP} + \boldsymbol{I}^{UP}\right)\boldsymbol{X}_0^{UP} \tag{13}$$

$$\tilde{\boldsymbol{X}}_L^{UP} = \prod_{\ell=1}^{L}\left(\boldsymbol{W}_\ell^{UP} + \Delta\boldsymbol{W}_\ell^{UP} + \boldsymbol{I}^{UP}\right)\boldsymbol{X}_0^{UP} \tag{14}$$

By taking difference between Eq. (13) and Eq. (14), we have Eq. (15).

$$\begin{aligned}\tilde{\boldsymbol{X}}_L^{UP} - \boldsymbol{X}_L^{UP} = \Bigg(&\prod_{\ell=1}^{L}\left(\boldsymbol{W}_\ell^{UP} + \Delta\boldsymbol{W}_\ell^{UP} + \boldsymbol{I}^{UP}\right) \\ &- \prod_{\ell=1}^{L}\left(\boldsymbol{W}_\ell^{UP} + \boldsymbol{I}^{UP}\right)\Bigg)\boldsymbol{X}_0^{UP}\end{aligned} \tag{15}$$

By assuming enough small $\Delta\boldsymbol{W}_\ell^{UP}$ as in Assumption C.2 and approximating the difference to first-order terms, we have Eq. (16).

$$\begin{aligned}&\tilde{\boldsymbol{X}}_L^{UP} - \boldsymbol{X}_L^{UP} \\ &\approx \sum_{\ell=1}^{L}\Delta\boldsymbol{W}_\ell^{UP}\prod_{k\neq\ell}^{L}\left(\boldsymbol{W}_k^{UP} + \boldsymbol{I}\right)\boldsymbol{X}_0^{UP}\end{aligned} \tag{16}$$

From the submultiplicative property, the norm of Eq. (16) can be bounded as Eq. (17)

$$\left\|\tilde{\boldsymbol{X}}_L^{UP} - \boldsymbol{X}_L^{UP}\right\| \leq L\epsilon(1+\delta)^{L-1}\left\|\boldsymbol{X}_0^{UP}\right\| \tag{17}$$

□

Next, we analyze the accumulated output shift bound of the width-upscaled residual network with function-preserving lower layers and non-preserving upper layers. A function-preserving layer has frozen $\boldsymbol{W}^{12}$ which is zero-initialized and a non-preserving layer has trainable $\boldsymbol{W}^{12}$. If the original function is preserved for L_{fp} lower layers, the $L - L_{fp}$ upper layers may contribute to forgetting, while allowing greater learning opportunities. With this intuition and Assumption C.5, we extend Proposition C.4 to Proposition 3.3.

Assumption C.5. (Smaller Function-Preserving Weight Norm than Non-Preserving Weight Norm). For function-preserving lower layers $1 \leq \ell \leq L_{fp}$, the norm of its upscaled weight matrix is bounded by a small constant $\delta_{fp} > 0$, i.e., $\left\|\boldsymbol{W}_{\ell}^{UP}\right\| \leq \delta_{fp}$. On the other hand, for non-preserving upper layers $L_{fp} < \ell \leq L$, the norm of its upscaled weight matrix is bounded by a small constant $\delta_{np} > 0$, i.e., $\left\|\boldsymbol{W}_{\ell}^{UP}\right\| \leq \delta_{np}$. Now, we assume that $\delta_{fp} < \delta_{np}$ since $\boldsymbol{W}^{12}$ is frozen as zero-initialized for δ_{fp} while $\boldsymbol{W}^{12}$ is trainable for δ_{np}.

Proposition 3.3. *The accumulated output shift of the width-upscaled residual network with function-preserving lower layers $1 \leq \ell \leq L_{fp}$ and non-preserving upper layers $L_{fp} < \ell \leq L$ is bounded by Eq. (18).*

$$\left\|\tilde{\boldsymbol{X}}_{L}^{UP} - \boldsymbol{X}_{L}^{UP}\right\| \leq (L - L_{fp})\epsilon(1+\delta_{np})^{L-1}\left(\frac{1+\delta_{fp}}{1+\delta_{np}}\right)^{L_{fp}}\left\|\boldsymbol{X}_{0}^{UP}\right\| \tag{18}$$

where $\tilde{\boldsymbol{X}}_{L}^{UP}$ denotes updated output of width-upscaled residual network as defined by Definition C.3 and δ_{fp} and δ_{np} denotes upper bound of norm of upscaled weight matrix for function-preserving lower layers and non-preserving upper layers, respectively, under Assumption C.5.

Proof. By taking difference between Eq. (13) and Eq. (14) for $L_{fp}+1 \leq \ell \leq L$, we have Eq. (19).

$$\tilde{\boldsymbol{X}}_{L}^{UP} - \boldsymbol{X}_{L}^{UP} = \left(\prod_{\ell=L_{fp}+1}^{L}(\boldsymbol{W}_{\ell}^{UP} + \Delta\boldsymbol{W}_{\ell}^{UP} + \boldsymbol{I}^{UP}) - \prod_{\ell=L_{fp}+1}^{L}(\boldsymbol{W}_{\ell}^{UP} + \boldsymbol{I}^{UP})\right)\boldsymbol{X}_{L_{fp}}^{UP} \tag{19}$$

By assuming enough small $\Delta\boldsymbol{W}_{\ell}^{UP}$ as in Assumption C.2 and approximating the difference to first-order terms, we have Eq. (20).

$$\begin{aligned}
&\tilde{\boldsymbol{X}}_{L}^{UP} - \boldsymbol{X}_{L}^{UP} \\
&\approx \sum_{\ell=L_{fp}+1}^{L}\Delta\boldsymbol{W}_{\ell}^{UP}\prod_{k=L_{fp}+1,k\neq\ell}^{L}(\boldsymbol{W}_{k}^{UP}+\boldsymbol{I})\boldsymbol{X}_{L_{fp}}^{UP} \\
&= \sum_{\ell=L_{fp}+1}^{L}\Delta\boldsymbol{W}_{\ell}^{UP}\prod_{k=L_{fp}+1,k\neq\ell}^{L}(\boldsymbol{W}_{k}^{UP}+\boldsymbol{I})\prod_{\ell=1}^{L_{fp}}(\boldsymbol{W}_{\ell}^{UP}+\boldsymbol{I})\boldsymbol{X}_{0}^{UP}
\end{aligned} \tag{20}$$

From the submultiplicative property, the norm of Eq. (20) can be bounded as Eq. (21)

$$\left\|\tilde{\boldsymbol{X}}_{L}^{UP} - \boldsymbol{X}_{L}^{UP}\right\| \leq (L - L_{fp})\epsilon(1+\delta_{np})^{L-1}\left(\frac{1+\delta_{fp}}{1+\delta_{np}}\right)^{L_{fp}}\left\|\boldsymbol{X}_{0}^{UP}\right\| \tag{21}$$

Note that, since $\delta_{fp} < \delta_{np}$ from Assumption C.5, the forgetting bound increases exponentially with decreasing L_{fp}. At its extremes, forgetting happens the most (Eq. (21) degenerates to Eq. (17)) if no layer preserves the original function, i.e., $L_{fp} = 0$, whereas the forgetting bound becomes 0 if every layer preserves the original function, i.e., $L_{fp} = L$.

This completes the proof. □

Corollary 3.4. *Forgetting increases exponentially with decreasing L_{fp}, the number of function preserving $\boldsymbol{W}_{\ell}^{12}$ blocks in lower layers $1 \leq \ell \leq L_{fp}$.*

Proof. Immediate from Proposition 3.3. □

D Convergence Analysis

In this section, we analyze convergence of standard *Continual Learning* (CL) and *Routing-based Continual Learning* induced by logit routing. Following common CL analyses, we bound the *task-to-global weight divergence* $\left\|\mathbf{w}_t^i - \mathbf{w}_t^G\right\|$ via the *task-to-global gradient divergence* $\left\|\nabla F^i(\mathbf{w}) - \nabla F^G(\mathbf{w})\right\|$.

D.1 Objectives with Logit Routing

Single shared parameters and a coordinate partition. Let $\mathbf{w}$ denote the *single* trainable parameter vector. We use a coordinate partition $\mathbf{w} = (\psi, \phi)$, where ϕ denotes the *interference coordinates* (e.g., blocks such as $\boldsymbol{W}^{12}$) and ψ denotes the remaining coordinates. This is only a notational partition of one shared $\mathbf{w}$.

Adapt vs. Preserve logits. For an input $\mathbf{x}$, define:

- **Adapt logits:** $\boldsymbol{Z}_A(\mathbf{w}; \mathbf{x})$.
- **Preserve logits:**

$$\boldsymbol{Z}_P(\mathbf{w}; \mathbf{x}) \triangleq \boldsymbol{Z}_A((\psi, \mathbf{0}); \mathbf{x}), \tag{22}$$

  i.e., the same computation with the interference coordinates masked to $\mathbf{0}$.

Logit router (stop-gradient / fixed router). Given a threshold τ, define a binary routing decision $r(\mathbf{x}) \in \{0, 1\}$. To enable standard smooth optimization analysis, we treat the router as fixed

with respect to $\mathbf{w}$ during optimization (e.g., computed from a detached forward pass or from a snapshot $\bar{\mathbf{w}}$ and held constant):

$$r(\mathbf{x}) \triangleq \mathbb{I}\Big[\cos\big(\boldsymbol{Z}_P(\bar{\mathbf{w}};\mathbf{x}),\ \boldsymbol{Z}_A(\bar{\mathbf{w}};\mathbf{x})\big) \le \tau\Big], \tag{23}$$

where $r(\mathbf{x}) = 0$ indicates routing to *preserve* and $r(\mathbf{x}) = 1$ indicates routing to *adapt*.

Task objectives. Let $\mathcal{L}(\boldsymbol{Z}, y)$ be the per-sample loss (e.g., negative log-likelihood). The standard (adapt-only) task objective is

$$F_A^i(\mathbf{w}) \triangleq \frac{1}{|\mathcal{D}^i|} \sum_{(\mathbf{x},y)\in\mathcal{D}^i} \mathcal{L}\big(\boldsymbol{Z}_A(\mathbf{w};\mathbf{x}), y\big). \tag{24}$$

The routed task objective is

$$F_R^i(\mathbf{w}) \triangleq \frac{1}{|\mathcal{D}^i|} \sum_{(\mathbf{x},y)\in\mathcal{D}^i} \Big((1-r(\mathbf{x}))\,\mathcal{L}\big(\boldsymbol{Z}_P(\mathbf{w};\mathbf{x}), y\big) + r(\mathbf{x})\,\mathcal{L}\big(\boldsymbol{Z}_A(\mathbf{w};\mathbf{x}), y\big)\Big). \tag{25}$$

Virtual global objectives. Let $\mathcal{D}^G \triangleq \cup_{1\le i\le N}\mathcal{D}^i$. Define

$$F_A^G(\mathbf{w}) \triangleq \frac{1}{|\mathcal{D}^G|} \sum_{(\mathbf{x},y)\in\mathcal{D}^G} \mathcal{L}\big(\boldsymbol{Z}_A(\mathbf{w};\mathbf{x}), y\big),$$

$$F_R^G(\mathbf{w}) \triangleq \frac{1}{|\mathcal{D}^G|} \sum_{(\mathbf{x},y)\in\mathcal{D}^G} \Big((1-r(\mathbf{x}))\,\mathcal{L}\big(\boldsymbol{Z}_P(\mathbf{w};\mathbf{x}), y\big) + r(\mathbf{x})\,\mathcal{L}\big(\boldsymbol{Z}_A(\mathbf{w};\mathbf{x}), y\big)\Big). \tag{26}$$

D.2 Training Dynamics and a Generic Divergence Recursion

For either method (A or R), we consider gradient descent updates on task i:

$$\mathbf{w}_t^i \triangleq \mathbf{w}_{t-1}^i - \eta\nabla F^i(\mathbf{w}_{t-1}^i), \qquad t \ge 1, \tag{27}$$

initialized at the task start by $\mathbf{w}_0^i \triangleq \mathbf{w}_T^{i-1}$. The virtual global iterate is

$$\mathbf{w}_t^G \triangleq \mathbf{w}_{t-1}^G - \eta\nabla F^G(\mathbf{w}_{t-1}^G), \qquad t \ge 1, \tag{28}$$

initialized at $\mathbf{w}_0^G \triangleq \mathbf{w}_0^0$, with (F^i, F^G) chosen consistently (either (F_A^i, F_A^G) or (F_R^i, F_R^G)).

Lemma D.1 (One-step task-to-global weight divergence bound)**.** *Assume F^G is β-smooth. Then for either method,*

$$\big\|\mathbf{w}_t^i - \mathbf{w}_t^G\big\| \le (1+\eta\beta)\big\|\mathbf{w}_{t-1}^i - \mathbf{w}_{t-1}^G\big\| + \eta\,\big\|\nabla F^i(\mathbf{w}_{t-1}^i) - \nabla F^G(\mathbf{w}_{t-1}^i)\big\|. \tag{29}$$

Proof. From Eq. (27) and Eq. (28),

$$\mathbf{w}_t^i - \mathbf{w}_t^G = \mathbf{w}_{t-1}^i - \mathbf{w}_{t-1}^G - \eta\big(\nabla F^i(\mathbf{w}_{t-1}^i) - \nabla F^G(\mathbf{w}_{t-1}^G)\big). \tag{30}$$

Add and subtract $\nabla F^G(\mathbf{w}_{t-1}^i)$, apply triangle inequality, and use β-smoothness:

$$\big\|\nabla F^G(\mathbf{w}_{t-1}^i) - \nabla F^G(\mathbf{w}_{t-1}^G)\big\| \le \beta\big\|\mathbf{w}_{t-1}^i - \mathbf{w}_{t-1}^G\big\|. \tag{31}$$

Substituting yields Eq. (29). □

Thus, it suffices to compare the task-to-global gradient divergence $\big\|\nabla F^i(\mathbf{w}) - \nabla F^G(\mathbf{w})\big\|$ between standard CL and routing-based CL.

D.3 Assumptions for Routed Optimization

Assumption D.2 (Smoothness under a fixed router)**.** For every task i, the objectives F_A^i, F_A^G are convex and β-smooth. For routed objectives, F_R^i, F_R^G are convex and β-smooth when the router $r(\mathbf{x})$ is treated as fixed (Assumption D.3).

Assumption D.3 (Fixed router (stop-gradient))**.** During optimization, $r(\mathbf{x})$ is treated as fixed with respect to $\mathbf{w}$ (e.g., computed from a detached forward pass or from a snapshot $\bar{\mathbf{w}}$ and held constant for the updates being analyzed).

Assumption D.4 (Bounded interference gradients)**.** There exists $G_\phi > 0$ such that for all $\mathbf{w}$ and all $(\mathbf{x}, y)$,

$$\big\|\nabla_\phi\mathcal{L}\big(\boldsymbol{Z}_A(\mathbf{w};\mathbf{x}), y\big)\big\| \le G_\phi.$$

Assumption D.5 (Router stability)**.** Define

$$p_i \triangleq \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}^i}[r(\mathbf{x}) = 1], \qquad p_G \triangleq \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}^G}[r(\mathbf{x}) = 1]. \tag{32}$$

Assume $p_i \le p_{\max} < 1$ and $|p_i - p_G| \le \varepsilon_p$ for all i.

Assumption D.6 (Conditional divergence not amplified on ϕ)**.** For all $\mathbf{w}$ and all tasks i,

$$\big\|\nabla_\phi F_A^i(\mathbf{w} \mid r=1) - \nabla_\phi F_A^G(\mathbf{w} \mid r=1)\big\| \le \big\|\nabla_\phi F_R^i(\mathbf{w}) - \nabla_\phi F_A^G(\mathbf{w})\big\|, \tag{33}$$

where $F(\mathbf{w} \mid r=1)$ denotes the objective restricted to samples satisfying $r(\mathbf{x}) = 1$.

Assumption D.7 (No increase of divergence on ψ)**.** For all $\mathbf{w}$ and all tasks i,

$$\big\|\nabla_\psi F_R^i(\mathbf{w}) - \nabla_\psi F_R^G(\mathbf{w})\big\| \le \big\|\nabla_\psi F_A^i(\mathbf{w}) - \nabla_\psi F_A^G(\mathbf{w})\big\|. \tag{34}$$

D.4 Interference-Gradient Gating by Logit Routing

Lemma D.8 (Preservation logits have zero ϕ-gradient)**.** *For all* $(\mathbf{x}, y)$ *and all* $\mathbf{w} = (\psi, \phi)$*,*

$$\nabla_\phi \mathcal{L}\big(\boldsymbol{Z}_P(\mathbf{w}; \mathbf{x}), y\big) = \mathbf{0}.$$

Consequently, under Assumption D.3,

$$\nabla_\phi F_R^i(\mathbf{w}) = \tfrac{1}{|\mathcal{D}^i|} \textstyle\sum_{(\mathbf{x},y)\in\mathcal{D}^i} r(\mathbf{x})\, \nabla_\phi \mathcal{L}\big(\boldsymbol{Z}_A(\mathbf{w}; \mathbf{x}), y\big). \tag{35}$$

Proof. By Eq. (22), $\boldsymbol{Z}_P(\mathbf{w}; \mathbf{x})$ does not depend on ϕ, hence its ϕ-gradient is zero. Differentiating Eq. (25) with a fixed router yields Eq. (35). □

Lemma D.9 (Contraction of task-to-global divergence on ϕ)**.** *Under Assumptions D.3–D.6 and D.4–D.5, for all* $\mathbf{w}$*,*

$$\begin{aligned}&\big\|\nabla_\phi F_R^i(\mathbf{w}) - \nabla_\phi F_R^G(\mathbf{w})\big\| \\ &\le p_{\max} \big\|\nabla_\phi F_A^i(\mathbf{w}) - \nabla_\phi F_A^G(\mathbf{w})\big\| \\ &\qquad + \varepsilon_p G_\phi.\end{aligned} \tag{36}$$

Proof. Let $g(\mathbf{x}, y; \mathbf{w}) \triangleq \nabla_\phi \mathcal{L}(\boldsymbol{Z}_A(\mathbf{w}; \mathbf{x}), y)$. By Lemma D.8,

$$\begin{aligned}\nabla_\phi F_R^i(\mathbf{w}) &= \mathbb{E}_{\mathcal{D}^i}[r(\mathbf{x}) g(\mathbf{x}, y; \mathbf{w})], \\ \nabla_\phi F_R^G(\mathbf{w}) &= \mathbb{E}_{\mathcal{D}^G}[r(\mathbf{x}) g(\mathbf{x}, y; \mathbf{w})].\end{aligned} \tag{37}$$

Write each as $p \cdot \mathbb{E}[g \mid r = 1]$:

$$\begin{aligned}\nabla_\phi F_R^i(\mathbf{w}) &= p_i\, \mathbb{E}_{\mathcal{D}^i}[g \mid r = 1], \\ \nabla_\phi F_R^G(\mathbf{w}) &= p_G\, \mathbb{E}_{\mathcal{D}^G}[g \mid r = 1].\end{aligned} \tag{38}$$

Then

$$\begin{aligned}&\big\|\nabla_\phi F_R^i - \nabla_\phi F_R^G\big\| \\ &\le p_i \|\mathbb{E}_{\mathcal{D}^i}[g \mid r = 1] - \mathbb{E}_{\mathcal{D}^G}[g \mid r = 1]\| \\ &\qquad + |p_i - p_G|\, \|\mathbb{E}_{\mathcal{D}^G}[g \mid r = 1]\|.\end{aligned} \tag{39}$$

By Assumption D.4, $\|\mathbb{E}_{\mathcal{D}^G}[g \mid r = 1]\| \le G_\phi$, so the second term is bounded by $\varepsilon_p G_\phi$ using Assumption D.5. For the first term, note that $\mathbb{E}_{\mathcal{D}^i}[g \mid r = 1] = \nabla_\phi F_A^i(\mathbf{w} \mid r = 1)$ and similarly for $\mathcal{D}^G$. Assumption D.6 and $p_i \le p_{\max}$ yield Eq. (36). □

D.5 Main Theorem: Routed Convergence Bound

Theorem 4.1. Under Assumptions D.3–D.7 (and router stability condition Eq. (45)), *Routing-based Continual Learning* with logit routing admits a no-worse (and typically tighter) bound on the task-to-global weight divergence than standard *Continual Learning*, by gating interference-coordinate updates.

Proof. Fix any $\mathbf{w} = (\psi, \phi)$ and define the task-to-global gradient divergences

$$\begin{aligned}\Delta_A(\mathbf{w}) &\triangleq \big\|\nabla F_A^i(\mathbf{w}) - \nabla F_A^G(\mathbf{w})\big\|, \\ \Delta_R(\mathbf{w}) &\triangleq \big\|\nabla F_R^i(\mathbf{w}) - \nabla F_R^G(\mathbf{w})\big\|.\end{aligned} \tag{40}$$

Since ψ and ϕ form a coordinate partition of $\mathbf{w}$, the squared norm decomposes as

$$\begin{aligned}&\Delta_R(\mathbf{w})^2 \\ &= \big\|\nabla_\psi F_R^i(\mathbf{w}) - \nabla_\psi F_R^G(\mathbf{w})\big\|^2 \\ &\quad + \big\|\nabla_\phi F_R^i(\mathbf{w}) - \nabla_\phi F_R^G(\mathbf{w})\big\|^2.\end{aligned} \tag{41}$$

By Assumption D.7, the ψ-term satisfies

$$\big\|\nabla_\psi F_R^i - \nabla_\psi F_R^G\big\| \le \big\|\nabla_\psi F_A^i - \nabla_\psi F_A^G\big\|.$$

By Lemma D.9, the ϕ-term satisfies

$$\begin{aligned}&\big\|\nabla_\phi F_R^i - \nabla_\phi F_R^G\big\| \\ &\le p_{\max} \big\|\nabla_\phi F_A^i - \nabla_\phi F_A^G\big\| + \varepsilon_p G_\phi.\end{aligned} \tag{42}$$

Therefore, for all $\mathbf{w}$,

$$\begin{aligned}\Delta_R(\mathbf{w})^2 &\le \big\|\nabla_\psi F_A^i(\mathbf{w}) - \nabla_\psi F_A^G(\mathbf{w})\big\|^2 \\ &\quad + \Big(p_{\max} \big\|\nabla_\phi F_A^i(\mathbf{w}) - \nabla_\phi F_A^G(\mathbf{w})\big\| + \varepsilon_p G_\phi\Big)^2.\end{aligned} \tag{43}$$

Next, note that the adapt-only divergence also decomposes:

$$\begin{aligned}&\Delta_A(\mathbf{w})^2 \\ &= \big\|\nabla_\psi F_A^i(\mathbf{w}) - \nabla_\psi F_A^G(\mathbf{w})\big\|^2 \\ &\quad + \big\|\nabla_\phi F_A^i(\mathbf{w}) - \nabla_\phi F_A^G(\mathbf{w})\big\|^2.\end{aligned} \tag{44}$$

Hence, whenever the router is stable enough that

$$\varepsilon_p G_\phi \le (1 - p_{\max}) \big\|\nabla_\phi F_A^i(\mathbf{w}) - \nabla_\phi F_A^G(\mathbf{w})\big\|, \tag{45}$$

we have

$$p_{\max}\, a + \varepsilon_p G_\phi \le a \quad \text{for} \quad a = \big\|\nabla_\phi F_A^i - \nabla_\phi F_A^G\big\|, \tag{46}$$

and thus Eq. (43) implies $\Delta_R(\mathbf{w}) \le \Delta_A(\mathbf{w})$. Moreover, if the inequality in Eq. (45) is strict and $a > 0$, then $\Delta_R(\mathbf{w}) < \Delta_A(\mathbf{w})$.

Finally, apply Lemma D.1. Both methods satisfy the recursion

$$\big\|\mathbf{w}_t^i - \mathbf{w}_t^G\big\| \le (1 + \eta\beta)\big\|\mathbf{w}_{t-1}^i - \mathbf{w}_{t-1}^G\big\| + \eta\, \Delta(\mathbf{w}_{t-1}^i), \tag{47}$$

with $\Delta = \Delta_A$ for standard CL and $\Delta = \Delta_R$ for routing-based CL. Under the condition in Eq. (45), the driving term $\Delta_R(\mathbf{w}_{t-1}^i)$ is no larger (and typically smaller) than $\Delta_A(\mathbf{w}_{t-1}^i)$, yielding a no-worse (and typically tighter) upper bound on $\big\|\mathbf{w}_t^i - \mathbf{w}_t^G\big\|$ after unrolling the recursion.

This completes the proof. □