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 more on scaling the right structure than on scaling parameters alone. We introduce SCALE, a width upscaling architecture that inserts lightweight expansion into linear modules while freezing all pre-trained parameters. This preserves the residual and attention topologies and increases capacity without perturbing the base model's original functionality. SCALE is guided by 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 selectively trains a subset of 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 mitigates the severe forgetting observed with depth expansion while still acquiring new knowledge. In continual pre-training on a Korean corpus, SCALE variants achieve less forgetting on English evaluations and competitive gains on Korean benchmarks, with these variants offering the best overall stability-plasticity trade-off. Accompanying analysis clarifies when preservation provably holds and why the interplay between preservation and adaptation stabilizes optimization compared to standard continual learning setups.

1 Introduction

The era of effortless gains from brute-force scaling of large language models (LLMs) is nearing its end. Recent discussions among leading AI researchers emphasize that further progress will depend less on adding parameters or data and more on scaling *the right structure* while preserving previously acquired knowledge (Sutskever 2024; LeCun 2025). This view redirects attention to *strategic architectural expansion*: approaches that enable continual pre-training (CPT) while preserving the pre-trained knowledge base. Classical continual learning (CL) methods—regularization, replay, or parameter isolation—help retain earlier knowledge but do not allocate new representational capacity for adaptation (Kirkpatrick et al. 2017; Rolnick et al. 2019). By contrast, function-preserving transformations such as Net2Net (Chen, Goodfellow, and Shlens 2015) and its successors show that structural expansion can increase capacity without disrupting the base model's original functionality. However, depth upscaling such as LLaMA Pro (Wu et al. 2024) often perturbs hidden representations and exhibits forgetting during CPT, indicating that architectural balance between preservation and adaptation must be managed more delicately.

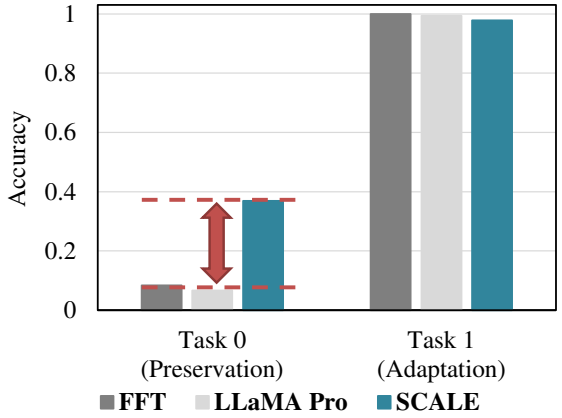


Figure 1: **Continual biography learning.**

We therefore propose SCALE (*upScaled ContinuAl LEarning*), a width upscaling architecture that introduces lightweight expansion inside linear modules while freezing all pre-trained parameters. SCALE enlarges the representational space of decoder-style LLMs without altering the residual topology or attention structure. Empirically, Figure 1 shows that depth-upscaled LLaMA Pro incurs severe forgetting on the *continual biography* task[1], whereas width-upscaled SCALE preserves prior knowledge significantly better while adapting to new knowledge. In CPT on the Korean dataset, Figure 2 highlights that SCALE achieves lower English forgetting perplexity and competitive Korean learning perplexity compared with representative baselines, in-

*These authors contributed equally.

†Corresponding author: sukhoon.jung@samsung.com

[1]Refer to Section 5 for details of continual biography task.

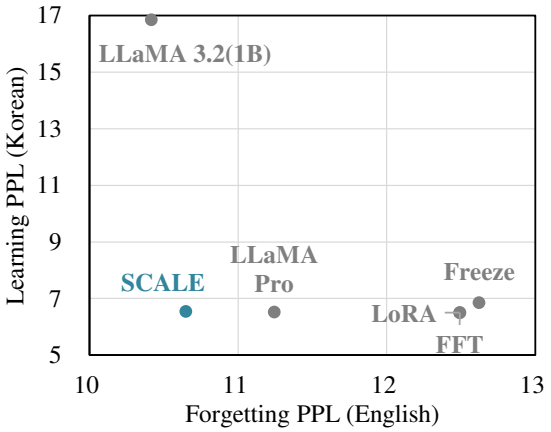


Figure 2: **Performance landscape.**

cluding LLaMA Pro and Freeze (Zheng et al. 2025), reflecting a superior stability–plasticity balance.

SCALE is instantiated by two complementary design principles developed empirically and supported theoretically: ① *Persistent Preservation* and ② *Collaborative Adaptation*. *Persistent Preservation* maintains the original function throughout training via preservation-oriented initialization and freezing patterns within the expansion. *Collaborative Adaptation* selectively trains a subset of expansion blocks (e.g., in upper layers or specific modules) to capture new domain knowledge while interacting stably with the frozen base. Our preliminary studies reveal that preservation-first configurations exhibit extreme resistance to forgetting, whereas collaborative configurations unlock strong adaptability; together they form a controllable frontier between stability and plasticity.

Based on the two principles, we introduce a family of width-upscaled learning methods: (i) SCALE-Preserve, which enforces preservation by maintaining the pre-trained function; (ii) SCALE-Adapt, which unlocks plasticity by collaborating with the frozen base; and (iii) SCALE-Route, which routes tokens between preservation and adaptation paths using a similarity-based router. Because adaptation paths can override preservation paths, SCALE-Route exposes both behaviors within a single forward pass and routes to the more relevant logits. We further derive a convergence advantage of routing-based CL over standard CL, theoretically supporting the observed stability–plasticity gains.

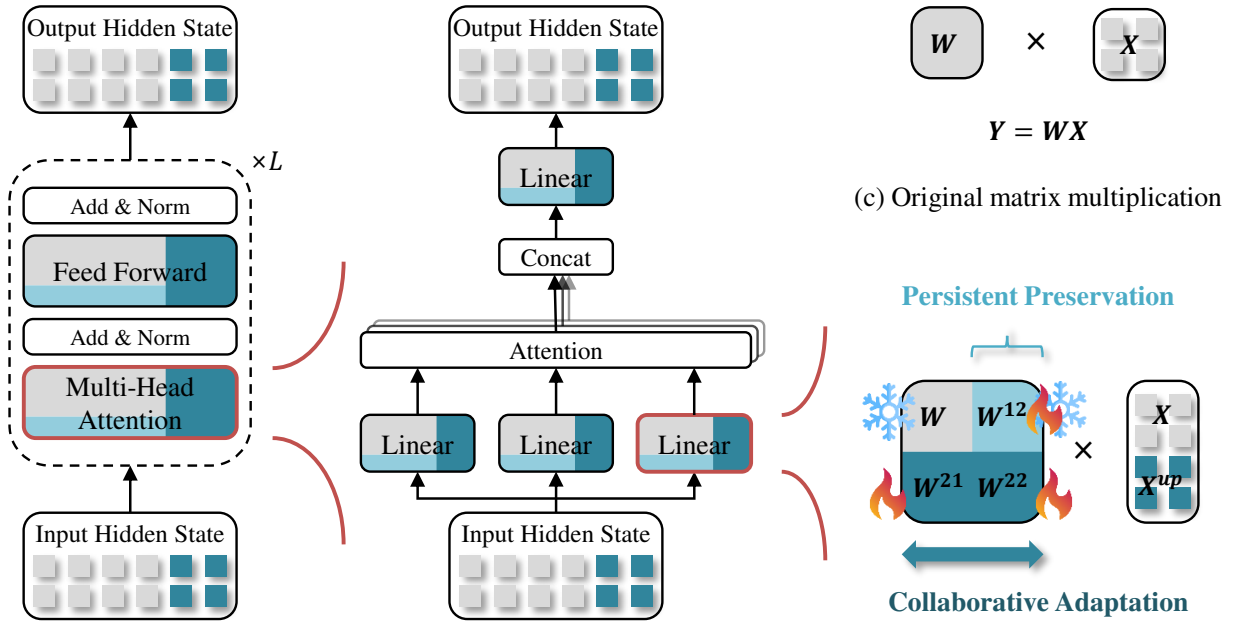
Key Contributions

1. *Width Upscaling Architecture.* SCALE freezes all pre-trained parameters and expands capacity with lightweight blocks inside linear modules, preserving the base computation graph and residual/attention structure (Section 3).
2. *Design Principles with Evidence.* We formalize *Persistent Preservation* and *Collaborative Adaptation*. Theoretical analyses and preliminary studies show persistent function preservation under the preservation-oriented setup and strong plasticity under the collaborative setup (Section 3).
3. *Width-Upscaled Learning Methods.* We introduce SCALE-Preserve (preservation-first), SCALE-Adapt (adaptation-first), and SCALE-Route (advantages-of-both). SCALE-Route routes tokens between preservation and adaptation paths, and enjoys a tighter convergence bound than standard CL (Section 4).
4. *Empirical Validation.* On a controlled biography task and CPT over the Korean dataset, SCALE variants reduce forgetting on English evaluations while maintaining or improving target-domain performance; SCALE-Route delivers the strongest stability–plasticity balance among the evaluated baselines (FFT, LoRA, Freeze, LLaMA Pro) under our settings (Section 5).

2 Related Work

Continual Learning Continual Learning (CL) aims to adapt models to sequentially arriving tasks without catastrophic forgetting of previously acquired knowledge. Classical CL approaches include regularization (Kirkpatrick et al. 2017; Zenke, Poole, and Ganguli 2017; Aljundi et al. 2018), replay (Rolnick et al. 2019; Shin et al. 2017; Sun, Ho, and Lee 2019), and parameter isolation methods (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 the extensive range of pre-trained knowledge while incorporating new linguistic or domain-specific knowledge. However, conventional CL approaches designed for smaller models or narrow-domain tasks often struggle when applied to LLMs, due to computational and privacy constraints. Recent studies emphasize parameter-efficient fine-tuning (PEFT) and architectural extensions that allocate new capacity for adaptation, enabling LLMs to retain general knowledge and improve performance in target domains such as multilingual settings or specialized industries.

Upscaled Learning Model upscaling aims to enhance the capabilities of pre-trained LLMs by efficiently expanding their architectures while preserving or improving performance. This approach can be broadly categorized into the types of upscaling: depth and width upscaling. Depth upscaling increases the number of layers to capture more complex representations while leveraging existing pre-trained weights. SOLAR (Kim et al. 2023) and LLaMA Pro (Wu et al. 2024) expand model depth through layer duplication and interleaving, followed by continual pre-training to recover performance. In contrast, width upscaling horizontally expands the model by increasing the hidden state dimension or the number of attention heads. Function-preserving transformations (Chen, Goodfellow, and Shlens 2015; Chen et al. 2021) enable width upscaling to maintain consistency with the original model outputs. Building on this idea, various width upscaling approaches (Shen et al. 2022; Yao et al. 2023; Samragh et al. 2024) along with depth upscaling initialize larger models from smaller pre-trained ones to minimize training overhead. From the perspective of continual learning, ELLE (Qin et al. 2022) and LOIRE (Han et al. 2025) apply both depth and width upscaling with function-preserving initialization to mitigate catastrophic forgetting while incorporating new domain-specific 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 propose novel width upscaling SCALE architecture, provide empirical findings, and corroborate them with theoretical analyses. Above all, an overview of SCALE architecture is presented in Section 3.1, followed by two design principles: ① *Persistent Preservation* and ② *Collaborative Adaptation*. Based on *Persistent Preservation* in Section 3.2, SCALE zero-initializes and freezes $\boldsymbol{W}^{12}$ to persistently preserve the original function until the end of training. In the meanwhile, based on *Collaborative Adaptation* in Section 3.3, SCALE collaboratively trains certain weight blocks to effectively adapt to new domain knowledge while preserving 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

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

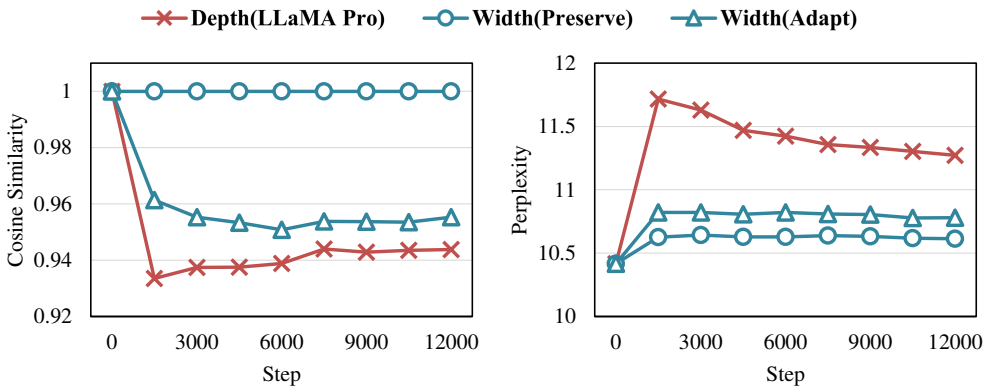
produce the corresponding query, key, and value representations for new heads. Consequently, SCALE has a structural constraint that all the weight matrices in MHA must be expanded such that the number of rows increases by an integer multiple of the head dimension. For embedding and output projection weight matrices, we upscale as follows:

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

3.2 Design Principle 1: *Persistent Preservation*

Appropriate weight block initialization and freeze strategy can prevent forgetting by allowing the original function $\boldsymbol{W}\boldsymbol{X}$ to be persistently preserved even until the end of training, thus called *Persistent Preservation*. We mainly insist that $\boldsymbol{W}^{12}$ be zero-initialized and frozen and also suggest that $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$ be initialized by imitating $\boldsymbol{W}$ and $\boldsymbol{W}^{12}$, respectively.

Initialization and Freeze of $\boldsymbol{W}^{12}$ As a preliminary study with regard to function preservation, we compare depth (LLaMA Pro (Wu et al. 2024)) and width (SCALE)



(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.**

upscaling from the perspective of how representation forgetting causes forgetting perplexity of pre-trained English knowledge[2], as shown in Figure 4. Specifically, Width(Preserve) is a basic setup of SCALE with zero-initialized $\boldsymbol{W}^{12}$ which is frozen for all layers for function preservation and the rest of randomly initialized trainable weights, whereas Width(Adapt) is a contrastive setup of SCALE including non-preserving $\boldsymbol{W}^{12}$ which is set to be trainable for all layers.

Above all, Figure 4a shows representation forgetting measured by cosine similarity of output hidden states after the last decoder layer between step 0 and subsequent training steps. We only compare the original part of the output hidden states because the depth upscaling does not involve expanding the hidden state dimension. As intended in the experimental design, Width(Preserve) never forgets the original function $\boldsymbol{WX}$, whereas Depth(LLaMA Pro) abruptly forgets around step 1000 and has a hard time recovering back for the subsequent steps. This resistance to forgetting stems from the amount of function preservation. The width upscaling with zero-initialized frozen $\boldsymbol{W}^{12}$ allows the original function to be persistently preserved until the end of training, whereas the depth upscaling inevitably perturbs the original function even at the beginning steps due to trainable upscaled layers in the middle. In contrast, Width(Adapt) gradually forgets representation as in Figure 4a and ends up exhibiting slightly higher forgetting perplexity than Width(Preserve), which is still significantly lower than Depth(LLaMA Pro), as in Figure 4b.

To corroborate our preliminary study, we analyze the function preservation of width upscaling in Theorem 3.1 and derive that setting $\boldsymbol{W}^{12}$ to $\mathbf{0}$ is necessary for all layers, which can be achieved by zero-initialization. Furthermore, from the difference between Width(Preserve) and Width(Adapt), we insist that freeze of $\boldsymbol{W}^{12}$ as well as zero-initialization be necessary for *Persistent Preservation*.

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

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

Proof. We defer proof to Appendix B for lack of space. □

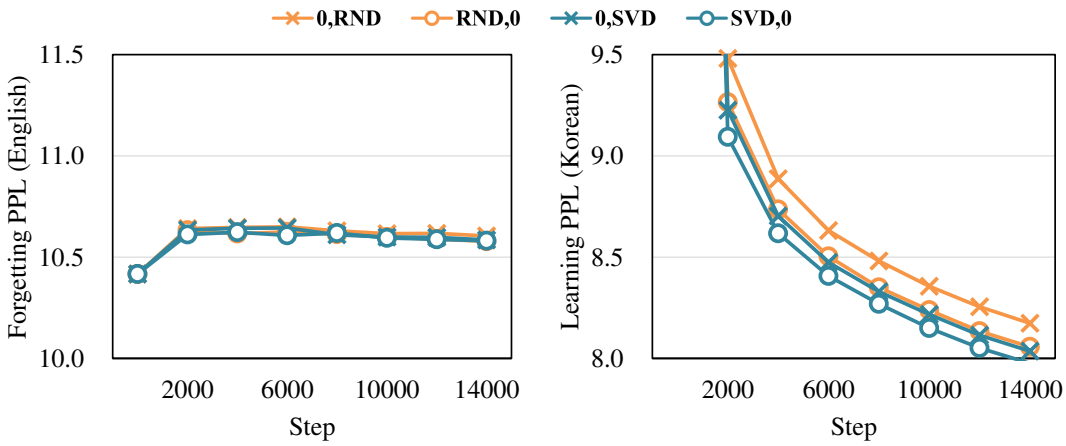
Initialization of $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$ We also compare initialization pairs for $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$ in Figure 5. **0** denotes zero-initialization, **RND** denotes random initialization (He et al. 2015), and **SVD** denotes dimension-reduced SVD of $\boldsymbol{W}$ to fit the dimensions of $\boldsymbol{W}^{21}$ or $\boldsymbol{W}^{22}$, suggested by LESA (Yang et al. 2025). Interestingly, every initialization pair exhibits near-perfect preservation in Figure 5a. On the other hand, adaptation performance consistently maintains the following order in Figure 5b: **0,RND** < **RND,0** < **0,SVD** < **SVD,0**. Therefore, we choose **SVD,0** as the default initialization strategy for $\boldsymbol{W}^{21}$ and $\boldsymbol{W}^{22}$ in this paper. Corollary 3.2 provides an analysis that supports this finding.

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 $\boldsymbol{W}_\ell^{22}$ values are irrelevant to the function preservation.*

3.3 Design Principle 2: *Collaborative Adaptation*

By collaboratively training certain weight blocks, SCALE can effectively adapt to new domain knowledge, thus called *Collaborative Adaptation*. We mainly suggest that weight blocks in certain layers or in certain modules be collaboratively trained.

Collaborative Layers We introduce a preliminary study to show a collaborative property of SCALE that leads to a trade-off of forgetting and learning. Since frozen $\boldsymbol{W}^{12}$ for the *Persistent Preservation* strictly prevents collaboration between $\boldsymbol{X}$ and $\boldsymbol{X}^{up}$, it inherently lacks learning capacity. Therefore, we initialize $\boldsymbol{W}^{12}$ to be $\mathbf{0}$ and permit certain amount of $\boldsymbol{W}^{12}$ blocks only in upper layers to be trainable, thus collaborating with the original weight block $\boldsymbol{W}$ in upper layers while $\boldsymbol{W}^{12}$ blocks in lower layers are still preserved. Figure 6 displays a trade-off of forgetting and learning PPL according to L_{fp} which is the number of function



(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}. A paired name of two methods is delimited by a comma.**

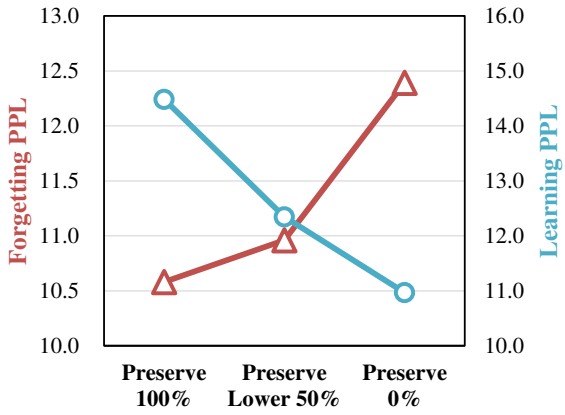


Figure 6: **Trade-off of forgetting and learning PPL according to the amount of preserving lower layers L_{fp}.**

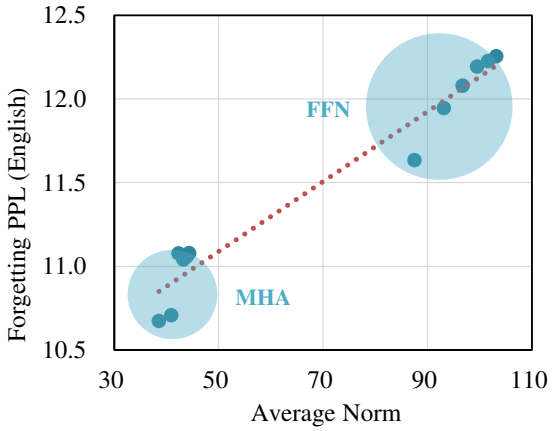


Figure 7: **Near-linear scalability of forgetting perplexity to upscaled weight norm.**

preserving $\boldsymbol{W}^{12}$ blocks in lower layers. It should be also noted that forgetting happens exponentially with decreasing L_{fp}. Proposition 3.3 and Corollary 3.4 provide an analysis that supports this finding.

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).

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

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 proofs in this section to Appendix C. □

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

Collaborative Modules In addition to collaborative layers, we further look into which module is more effective in preservation and collaboration. Specifically, based on function-preserving lower half layers, i.e., $L_{fp} = L/2$, we train two cases for an epoch with $\boldsymbol{W}^{12}$ blocks in either MHA or FFN module to be trainable, thus collaborating with $\boldsymbol{W}$ blocks only in the module's upscaled weights. Figure 7 shows that both cases in total exhibit near-linear scalability of forgetting perplexity to average norm of all upscaled weights. The case with FFN module exhibits high forgetting perplexity due to its large intermediate dimension size and, considering baseline forgetting perplexity of LLaMA-3.2-1B around 10.4, the converged perplexity suggests it is

not a good choice. On the other hand, the other case with MHA module converges within smaller error bound and exhibits only slightly higher perplexity compared to the baseline, which makes it a good choice for collaborative module.

4 Proposed Learning Methods

Persistent Preservation and *Collaborative Adaptation*, two design principles of SCALE, suggest complementary methods for upscaled continual learning. Additionally, taking only the advantages of both methods could possibly improve preservation and adaptation simultaneously. To this end, we propose three novel learning methods: ① SCALE-Preserve, ② SCALE-Adapt, and ③ SCALE-Route.

- SCALE-Preserve is the preservation-first method such that $\forall \ell$, $\boldsymbol{W}_{\ell}^{12}$ is initialized to $\mathbf{0}$ and not trainable, as defined by 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, on the other hand, is the adaptation-first method such that $\forall \ell$, $\boldsymbol{W}_{\ell}^{12}$ is initialized to $\mathbf{0}$ and trainable, as defined by Eq. (4).

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

- SCALE-Route is the advantages-of-both method that aims at maximizing preservation and adaptation simultaneously and thus mitigating the performance trade-off illustrated in Figure 6. Note that the core difference between preservation and adaptation of SCALE lies in trainability of the upscaled weights $\boldsymbol{W}^{12}$. Therefore, delicate utilization of trainable computation paths is key to the optimal performance. SCALE-Route routes the best computation paths to either SCALE-Preserve or SCALE-Adapt, as defined by Eq. (5).

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

  where cosine$(\cdot)$ denotes cosine similarity between two logits, which acts as a router controlled by minimum threshold τ. In brief, if logits of a token for $\boldsymbol{Z}_{preserve}$ and $\boldsymbol{Z}_{adapt}$ are similar, it routes the token to SCALE-Preserve. Otherwise, it takes an adaptation opportunity by routing the token to SCALE-Adapt. $\boldsymbol{Z}_{preserve}$ is approximated by $(\boldsymbol{Z}_{preserve} + \boldsymbol{Z}_{adapt})/2$ due to better trainability of $\boldsymbol{Z}_{adapt}$. Note that according to Section 3.2, the computation paths of SCALE-Adapt override those of SCALE-Preserve, thereby allowing SCALE-Adapt to produce logits of SCALE-Preserve as long as $\boldsymbol{W}^{12}$ being computed as $\mathbf{0}$ matrix. Therefore, it still requires a single forward pass with slight extra computation to produce both logits at any training step.

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

Model	Learning rate
FFT	1×10^{-5}
LoRA	1×10^{-5}
Freeze	1×10^{-5}
LLaMA Pro	2×10^{-4}
SCALE	1×10^{-3}

Table 1: **Learning rates of continual pre-training on the Korean dataset.**

Theorem 4.1. Routing-based Continual Learning *achieves lower convergence than standard* Continual Learning.

Proof. The proof is deferred to Appendix D. □

5 Experiments

5.1 Experimental Setup

The Biography Dataset We reproduce the controlled experiment conducted in (Zheng et al. 2025) to compare forgetting between existing continual learning methods and our proposed methods, especially SCALE-Route. The experimental dataset, *Biography Dataset*[3], consists of 200,000 synthetic individuals, each characterized by a name and six attributes: birthday, birth city, university attended, major, company name, and company city. For each individual, the dataset is divided into pre-training and fine-tuning data. As in (Zheng et al. 2025), our continual learning setting is constructed through three stages. The model is initially pre-trained on the first 100,000 individuals of the synthetic Biography Dataset, followed by fine-tuning on QA data corresponding to the first 50,000 individuals (Task 0). We then apply an upscaling method and fine-tune new QA data from a previously unseen 20,000 individuals (Task 1). During learning Task 1, we monitor the decline of the model's performance on Task 0, using *hard first-token accuracy*, a metric that measures whether the model's top-predicted first token matches the correct token.

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 the per-

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

formance 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, 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. As shown in Table 1, 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. For Freeze, we employ the same learning rate of FFT, as in the original paper. 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 has the lower increase of perplexity on the English test data in early stage of training, and therefore it can be regarded as a more stable approach for Continual Pre-Training. We also observe that the discrepancy of the perplexity between SCALE-Preserve and SCALE-Adapt arises more from learning than from forgetting, proving that training $\boldsymbol{W}^{12}$ more strongly affects learning than forgetting.

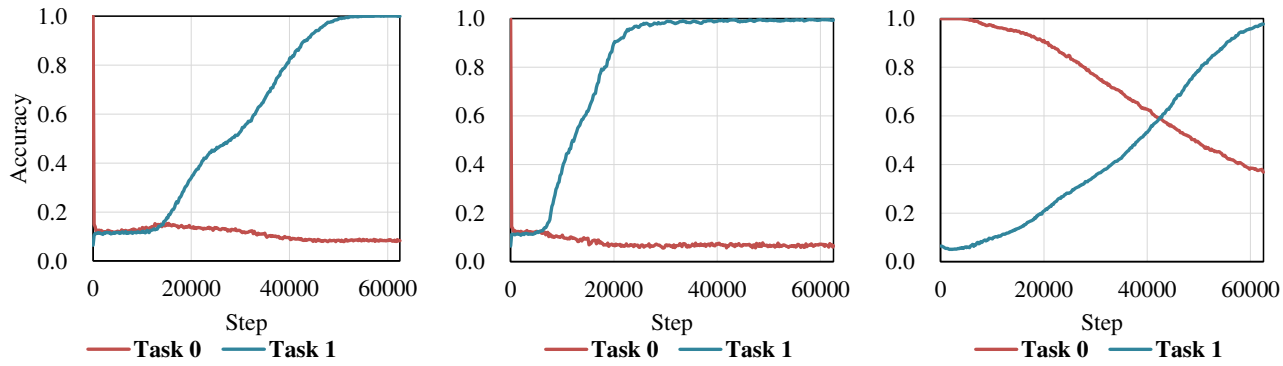
Furthermore, we evaluate our SCALE methods with the English and Korean benchmarks. Evaluations are conducted using Eleuther AI Language Model Evaluation Harness[4] and in a zero shot setting. Due to the lack of instruction-following capability of pre-trained models, we select only a very limited set of Korean benchmarks, KoBEST (Jang et al. 2022), for evaluation. The results are presented in Table 2. 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.

6 Conclusion

This work presented SCALE, a practical recipe for architectural expansion that enables stable continual pre-training without altering a model's core computation graph. By growing width inside linear submodules and freezing the original parameters, SCALE adds representational capacity while keeping the pre-trained function intact. Two design principles—*Persistent Preservation* and *Collaborative Adaptation*—let practitioners dial in the desired balance: preservation-oriented setups deliver extreme resistance to forgetting, while collaborative setups unlock strong plasticity by training a targeted subset of expansion components, preferably in upper layers and within attention when stability is paramount.

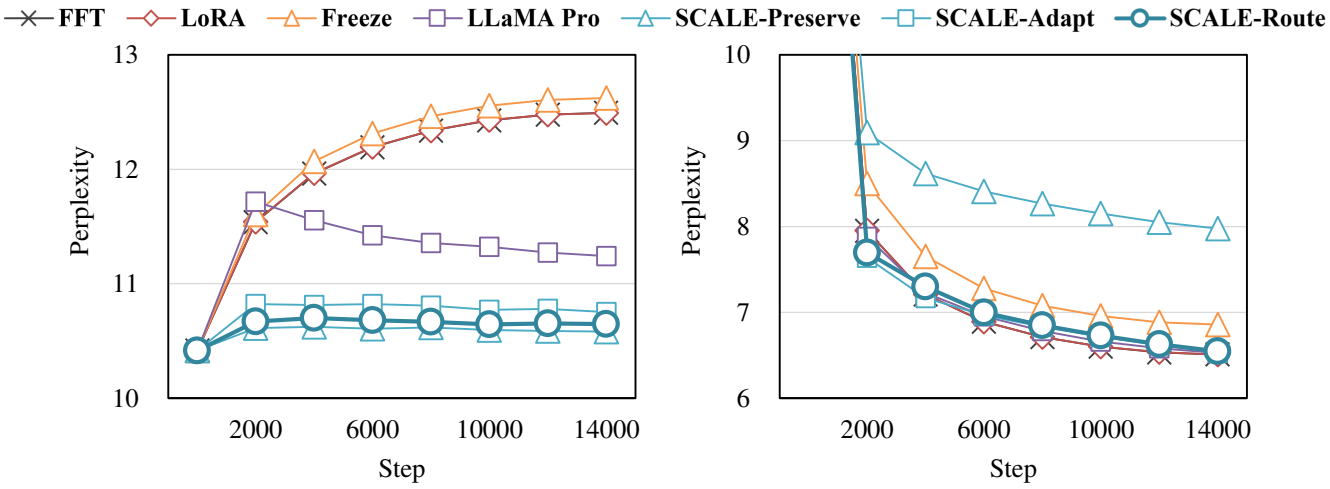
We instantiated these principles through three learning variants. SCALE-Preserve offers a preservation-first baseline with strong stability. SCALE-Adapt emphasizes plasticity by enabling collaboration throughout the network. SCALE-Route combines both by routing tokens between preservation and adaptation paths using a similarity criterion, exposing both behaviors in a single forward pass with small overhead. Empirically, width upscaling outperforms depth expansion on the biography task, achieving significantly higher retention and competitive adaptation. In continual pre-training on Korean web data, SCALE variants re-

[4]https://github.com/EleutherAI/lm-evaluation-harness



(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 the Korean subset of FineWeb2 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 2: **Performance comparison of upscaling methods trained on the Korean subset of FineWeb2 for one epoch.**

duce forgetting on English evaluations while matching or improving target-language learning; among them, SCALE-Route consistently achieves the best stability–plasticity balance. Theoretical results clarify why preservation holds under our initialization and freezing scheme and why routing confers a convergence advantage.

Limitations include evaluation at modest scales and on a constrained set of domains, a simple static routing threshold, and conservative choices about which layers and modules collaborate. Future work includes scaling to larger backbones and longer training horizons, adaptive selection of collaborative layers and modules, richer routing policies, integration with parameter-efficient tuning and retrieval, and broader multilingual and domain-specialized CPT. Together, these directions position width-upscaled continual learning as a reliable path beyond brute-force scaling.

References

Ahn, K.; Cheng, X.; Song, M.; Yun, C.; Jadbabaie, A.; and Sra, S. 2023. Linear attention is (maybe) all you need (to understand transformer optimization). *arXiv preprint arXiv:2310.01082*.

Aljundi, R.; Babiloni, F.; Elhoseiny, M.; Rohrbach, M.; and Tuytelaars, T. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, 139–154.

Andonian, A.; Anthony, Q.; Biderman, S.; Black, S.; Gali, P.; Gao, L.; Hallahan, E.; Levy-Kramer, J.; Leahy, C.; Nestler, L.; Parker, K.; Pieler, M.; Phang, J.; Purohit, S.; Schoelkopf, H.; Stander, D.; Songz, T.; Tigges, C.; Thérien, B.; Wang, P.; and Weinbach, S. 2023. GPT-NeoX: Large Scale Autoregressive Language Modeling in PyTorch. https://www.github.com/eleutherai/gpt-neox.

Biderman, S.; Schoelkopf, H.; Anthony, Q. G.; Bradley, H.; O'Brien, K.; Hallahan, E.; Khan, M. A.; Purohit, S.; Prashanth, U. S.; Raff, E.; et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, 2397–2430. PMLR.

Black, S.; Biderman, S.; Hallahan, E.; Anthony, Q.; Gao, L.; Golding, L.; He, H.; Leahy, C.; McDonell, K.; Phang, J.; et al. 2022. Gpt-neox-20b: An open-source autoregressive language model. *arXiv preprint arXiv:2204.06745*.

Boix-Adsera, E.; Littwin, E.; Abbe, E.; Bengio, S.; and Susskind, J. 2023. Transformers learn through gradual rank increase. *Advances in Neural Information Processing Systems*, 36: 24519–24551.

Chen, C.; Yin, Y.; Shang, L.; Jiang, X.; Qin, Y.; Wang, F.; Wang, Z.; Chen, X.; Liu, Z.; and Liu, Q. 2021. bert2bert: Towards reusable pretrained language models. *arXiv preprint arXiv:2110.07143*.

Chen, T.; Goodfellow, I.; and Shlens, J. 2015. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*.

Deb, M.; and Ogunfunmi, T. 2025. Information-Theoretical Analysis of a Transformer-Based Generative AI Model. *Entropy*, 27(6): 589.

Du, W.; Luo, T.; Qiu, Z.; Huang, Z.; Shen, Y.; Cheng, R.; Guo, Y.; and Fu, J. 2024. Stacking Your Transformers: A Closer Look at Model Growth for Efficient LLM Pre-Training. *arXiv preprint arXiv:2405.15319*.

Han, X.; Wang, Y.; Feng, J.; Hu, Q.; Deng, C.; et al. 2025. LOIRE: LifelOng learning on Incremental data via pre-trained language model gRowth Efficiently. In *The Thirteenth International Conference on Learning Representations*.

He, B.; Martens, J.; Zhang, G.; Botev, A.; Brock, A.; Smith, S. L.; and Teh, Y. W. 2023. Deep transformers without shortcuts: Modifying self-attention for faithful signal propagation. *arXiv preprint arXiv:2302.10322*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2): 3.

Jang, M.; Kim, D.; Kwon, D. S.; and Davis, E. 2022. Kobest: Korean balanced evaluation of significant tasks. In *Proceedings of the 29th International Conference on Computational Linguistics*, 3697–3708.

Kim, D.; Park, C.; Kim, S.; Lee, W.; Song, W.; Kim, Y.; Kim, H.; Kim, Y.; Lee, H.; Kim, J.; et al. 2023. Solar 10.7 b: Scaling large language models with simple yet effective depth up-scaling. *arXiv preprint arXiv:2312.15166*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

LeCun, Y. 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".

Li, H.; Wang, M.; Liu, S.; and Chen, P.-Y. 2023. A theoretical understanding of shallow vision transformers: Learning, generalization, and sample complexity. *arXiv preprint arXiv:2302.06015*.

Li, X. L.; and Liang, P. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Liu, L.; Zhang, J.; Song, S.; and Letaief, K. B. 2019. Edge-assisted hierarchical federated learning with non-iid Data. *arXiv:1905.06641*.

Nguyen, T. Q.; and Salazar, J. 2019. Transformers without tears: Improving the normalization of self-attention. *arXiv preprint arXiv:1910.05895*.

Penedo, G.; Kydlívcek, H.; Sabolvcec, V.; Messmer, B.; Foroutan, N.; Kargaran, A. H.; Raffel, C.; Jaggi, M.; Werra, L. V.; and Wolf, T. 2025. FineWeb2: One Pipeline to Scale Them All – Adapting Pre-Training Data Processing to Every Language. arXiv:2506.20920.

Qin, Y.; Zhang, J.; Lin, Y.; Liu, Z.; Li, P.; Sun, M.; and Zhou, J. 2022. Elle: Efficient lifelong pre-training for emerging data. *arXiv preprint arXiv:2203.06311*.

Qin, Z.; Zhou, J.; and Zhu, Z. 2025. On the Convergence of Gradient Descent on Learning Transformers with Residual Connections. *arXiv preprint arXiv:2506.05249*.

Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.; and Wayne, G. 2019. Experience replay for continual learning. *Advances in neural information processing systems*, 32.

Rusu, A. A.; Rabinowitz, N. C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; and Hadsell, R. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Samragh, M.; Mirzadeh, I.; Vahid, K. A.; Faghri, F.; Cho, M.; Nabi, M.; Naik, D.; and Farajtabar, M. 2024. Scaling smart: Accelerating large language model pre-training with small model initialization. *arXiv preprint arXiv:2409.12903*.

Shen, S.; Walsh, P.; Keutzer, K.; Dodge, J.; Peters, M.; and Beltagy, I. 2022. Staged training for transformer language models. In *International Conference on Machine Learning*, 19893–19908. PMLR.

Shin, H.; Lee, J. K.; Kim, J.; and Kim, J. 2017. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30.

Sun, F.-K.; Ho, C.-H.; and Lee, H.-Y. 2019. Lamol: Language modeling for lifelong language learning. *arXiv preprint arXiv:1909.03329*.

Sutskever, I. 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".

Wang, B. 2021. Mesh-Transformer-JAX: model-parallel implementation of transformer language model with JAX.

Wang, P.; Panda, R.; Hennigen, L. T.; Greengard, P.; Karlinsky, L.; Feris, R.; Cox, D. D.; Wang, Z.; and Kim, Y. 2023. Learning to grow pretrained models for efficient transformer training. *arXiv preprint arXiv:2303.00980*.

Wang, S.; Tuor, T.; Salonidis, T.; Leung, K. K.; Makaya, C.; He, T.; and Chan, K. 2019. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 37(6): 1205–1221.

Wu, C.; Gan, Y.; Ge, Y.; Lu, Z.; Wang, J.; Feng, Y.; Shan, Y.; and Luo, P. 2024. LLaMA Pro: Progressive LLaMA with Block Expansion. *arXiv preprint arXiv:2401.02415*.

Yang, Y.; Cao, Z.; Ma, X.; Yao, Y.; Qin, L.; Chen, Z.; and Zhao, H. 2025. LESA: Learnable LLM Layer Scaling-Up. *arXiv preprint arXiv:2502.13794*.

Yao, Y.; Zhang, Z.; Li, J.; and Wang, Y. 2023. Masked structural growth for 2x faster language model pre-training. *arXiv preprint arXiv:2305.02869*.

Zenke, F.; Poole, B.; and Ganguli, S. 2017. Continual learning through synaptic intelligence. In *International conference on machine learning*, 3987–3995. PMLR.

Zhai, S.; Talbott, W.; Srivastava, N.; Huang, C.; Goh, H.; Zhang, R.; and Susskind, J. 2021. An attention free transformer. *arXiv preprint arXiv:2105.14103*.

Zhang, R.; Han, J.; Liu, C.; Gao, P.; Zhou, A.; Hu, X.; Yan, S.; Lu, P.; Li, H.; and Qiao, Y. 2023. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*.

Zheng, J.; Cai, X.; Qiu, S.; and Ma, Q. 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, Zhou, and Zhu 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)

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

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 $\mathbf{0}$, as defined in Definition B.1.

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

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

Theorem 3.1. *Width-upscaled network with $\boldsymbol{W}_\ell^{12}$ set to $\mathbf{0}$ 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 $\mathbf{0}$, 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 $\pi_{\boldsymbol{Y}}\big(\mathcal{F}\big(\boldsymbol{X}_{\ell-1}^{UP}\big)\big) = F(\boldsymbol{X}_{\ell-1})$.

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

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 $\mathbf{0}$ 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., $\big\|\boldsymbol{W}_\ell^{UP}\big\| \le \delta$.

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., $\big\|\Delta\boldsymbol{W}_\ell^{UP}\big\| \le \epsilon$.

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).

$$\tilde{\boldsymbol{X}}_\ell^{UP} \triangleq \big(\boldsymbol{W}_\ell^{UP} + \Delta\boldsymbol{W}_\ell^{UP} + \boldsymbol{I}^{UP}\big)\tilde{\boldsymbol{X}}_{\ell-1}^{UP} \quad (1 \le \ell \le L) \tag{11}$$

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 $1 \le \ell \le L$ of the residual network with width upscaling is bounded by Eq.* (12)*.*

$$\Big\|\tilde{\boldsymbol{X}}_L^{UP} - \boldsymbol{X}_L^{UP}\Big\| \le L\epsilon(1+\delta)^{L-1}\big\|\boldsymbol{X}_0^{UP}\big\| \tag{12}$$

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.

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

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

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

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

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).

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

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

$$\Big\|\tilde{\boldsymbol{X}}_L^{UP} - \boldsymbol{X}_L^{UP}\Big\| \le L\epsilon(1+\delta)^{L-1}\big\|\boldsymbol{X}_0^{UP}\big\| \tag{17}$$

□

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).

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

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).

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

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}, 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* and *Routing-based Continual Learning*. First of all, we define *Continual Learning* and *Routing-based Continual Learning* formally and make the following assumptions for the i-th task loss function F^i, as in many other relevant studies (Liu et al. 2019; Wang et al. 2019).

Definition D.1. (Continual Learning) Given a sequence of tasks $\mathcal{T} = \left\{T^1, T^2, \cdots, T^N\right\}$, each task T^i involves a set of data examples $(\mathbf{x}, y) \in \mathcal{D}^i$ and loss of predictions $F^i(\mathbf{w}) \triangleq \sum_{(\mathbf{x},y)\in\mathcal{D}^i}\frac{1}{|\mathcal{D}^i|}\mathcal{L}(\mathbf{w}, \mathbf{x}, y)$ with a loss function $\mathcal{L}$ parameterized by $\mathbf{w}$ over $\mathcal{D}^i$. If all task data could be accessed at once, the ideal continual learning objective would be defined by Eq. (22).

$$\mathbf{w}^* = \arg\min_{\mathbf{w}}\frac{1}{N}\sum_{i=1}^{N}F^i(\mathbf{w}) \tag{22}$$

Definition D.2. (Routing-based Continual Learning) Routing-based continual learning extends Definition D.1 by routing each data sample to the most relevant group weights among multiple candidate groups such that it achieves smaller loss of predictions.

Assumption D.3. For every task i, (1) F^i is convex; (2) F^i is ρ-Lipschitz, i.e., $\left\|F^i(\mathbf{w}) - F^i(\mathbf{w}')\right\| \leq \rho\|\mathbf{w} - \mathbf{w}'\|$ for any $\mathbf{w}$ and $\mathbf{w}'$; and (3) F^i is β-smooth, i.e., $\left\|\nabla F^i(\mathbf{w}) - \nabla F^i(\mathbf{w}')\right\| \leq \beta\|\mathbf{w} - \mathbf{w}'\|$ for any $\mathbf{w}$ and $\mathbf{w}'$.

Next, we analyze the convergence bound of standard *Continual Learning* from the perspective of task weight divergence and extend it to *Routing-based Continual Learning*. We start by expressing task weight update and defining virtual global task and task weight divergence. Starting from the previous task weights $\mathbf{w}^{i-1}, 1 \leq i \leq N$, i-th task weight update can be expressed as Eq. (23). w_0^0 is the very initial weights like a randomly initialized one or a pre-trained one.

$$\mathbf{w}_t^i \triangleq \begin{cases}\mathbf{w}_t^{i-1} & \text{if } t = 0 \\ \mathbf{w}_{t-1}^i - \eta\nabla F^i\left(\mathbf{w}_{t-1}^i\right) & \text{if } t > 0\end{cases} \tag{23}$$

Definition D.4. (Virtual Global Task) We define a virtually aggregated global task such that it consists of datasets of all tasks $\mathcal{D}^G \triangleq \cup_{1\leq i\leq N}\mathcal{D}^i$. Accordingly, global task loss can be expressed as $F^G\left(\mathbf{w}^G\right) \triangleq \sum_{(\mathbf{x},y)\in\mathcal{D}^G}\frac{1}{|\mathcal{D}^G|}\mathcal{L}\left(\mathbf{w}^G, \mathbf{x}, y\right)$.

Starting from the same aforementioned initial weights w_0^0, $\mathbf{w}^G$ is updated by Eq. (24).

$$\mathbf{w}_t^G \triangleq \begin{cases} \mathbf{w}_t^0 & \text{if } t = 0 \\ \mathbf{w}_{t-1}^G - \eta \nabla F^G\big(\mathbf{w}_{t-1}^G\big) & \text{if } t > 0 \end{cases} \tag{24}$$

Theorem 4.1. Routing-based Continual Learning *achieves lower convergence than standard* Continual Learning.

Proof. Firstly, we model a continual learning error by bounding norm of task weight divergence $\big\|\mathbf{w}_t^i - \mathbf{w}_t^G\big\|$.

$$\begin{aligned}
&\big\|\mathbf{w}_t^i - \mathbf{w}_t^G\big\| \\
&= \big\|\mathbf{w}_{t-1}^i - \eta \nabla F^i\big(\mathbf{w}_{t-1}^i\big) - \mathbf{w}_{t-1}^G + \eta \nabla F^G\big(\mathbf{w}_{t-1}^G\big)\big\| \\
&\qquad \text{(from Eq. (23) and Eq. (24))} \\
&= \|\mathbf{w}_{t-1}^i - \mathbf{w}_{t-1}^G - \eta(\nabla F^G\big(\mathbf{w}_{t-1}^i\big) - \nabla F^G\big(\mathbf{w}_{t-1}^G\big) \\
&\qquad + \nabla F^i\big(\mathbf{w}_{t-1}^i\big) - \nabla F^G\big(\mathbf{w}_{t-1}^i\big))\| \\
&\qquad \text{(adding a zero term)} \\
&\leq \big\|\mathbf{w}_{t-1}^i - \mathbf{w}_{t-1}^G\big\| + \eta\big\|\nabla F^G\big(\mathbf{w}_{t-1}^i\big) - \nabla F^G\big(\mathbf{w}_{t-1}^G\big)\big\| \\
&\qquad + \eta\big\|\nabla F^i\big(\mathbf{w}_{t-1}^i\big) - \nabla F^G\big(\mathbf{w}_{t-1}^i\big)\big\| \\
&\qquad \text{(from triangle inequality)} \\
&\leq (\eta\beta + 1)\big\|\mathbf{w}_{t-1}^i - \mathbf{w}_{t-1}^G\big\| \\
&\qquad + \eta\big\|\nabla F^i\big(\mathbf{w}_{t-1}^i\big) - \nabla F^G\big(\mathbf{w}_{t-1}^i\big)\big\| \\
&\qquad \text{(from } \beta\text{-smoothness of } F^G\text{)}
\end{aligned}$$

To conclude this proof, because this is a recursive inequality, it suffices to show that, in *Routing-based Continual Learning*, the last term $\big\|\nabla F^i(\mathbf{w}) - \nabla F^G(\mathbf{w})\big\|$ becomes smaller for any weights $\mathbf{w}$.

We assume that the negative log likelihood loss is used for any j-th data example and any weights $\mathbf{w}$, which can be expressed as Eq. (25).

$$\mathcal{L}(\mathbf{w}, \mathbf{x}_j, y_j) = -\sum_{c=1}^{C} \mathbb{I}_{jc} \log s_{jc}(\mathbf{w}), \tag{25}$$

where $\mathbb{I}_{jc} \triangleq \mathbb{I}(y_j = c)$ and $s_{jc}(\mathbf{w}) \triangleq s(\mathbf{w}, \mathbf{x}_j, y_j = c)$ denote the ground truth indicator and the softmax value, respectively, on the j-th data example belonging to the class c. Then, we can reformulate the task loss as Eq. (26).

$$\begin{aligned}
F^i(\mathbf{w}) &= -\sum_{j \in \mathcal{D}^i} \frac{1}{|\mathcal{D}^i|} \sum_{c=1}^{C} \mathbb{I}_{jc} \log s_{jc}(\mathbf{w}) \\
&= -\sum_{c=1}^{C} \frac{|\mathcal{D}^{ic}|}{|\mathcal{D}^i|} \sum_{j \in \mathcal{D}^{ic}} \frac{1}{|\mathcal{D}^{ic}|} \log s_{jc}(\mathbf{w}) \\
&= \sum_{c=1}^{C} \mathbb{P}\big(y_j = c | j \in \mathcal{D}^i\big) \mathbb{E}_{\mathcal{D}^{ic}}[-\log s_{jc}(\mathbf{w})] \\
&= \sum_{c=1}^{C} \mathbb{P}_{ic} \mathbb{E}_{\mathcal{D}^{ic}} \quad \text{(for brevity)}
\end{aligned} \tag{26}$$

where $\mathbb{E}_{\mathcal{D}^{ic}}$ denotes the expectation over $\mathcal{D}^{ic}$, which is the set of data examples belonging to the class c of the i-th task and can be defined as $\mathcal{D}^{ic} \triangleq \big\{j \in \mathcal{D}^i | y_j = c\big\}$. Based on the similar deduction, global task loss $F^G(\mathbf{w})$ can be reformulated as Eq. (27).

$$\begin{aligned}
F^G(\mathbf{w}) &= \sum_{c=1}^{C} \mathbb{P}\big(y_j = c | j \in \mathcal{D}^G\big) \mathbb{E}_{\mathcal{D}^{Gc}}[-\log s_{jc}(\mathbf{w})] \\
&= \sum_{c=1}^{C} \mathbb{P}_{Gc} \mathbb{E}_{\mathcal{D}^{Gc}} \quad \text{(for brevity)}
\end{aligned} \tag{27}$$

where $\mathcal{D}^{Gc} \triangleq \big\{j \in \mathcal{D}^G | y_j = c\big\}$.

Then, we obtain Eq. (28)

$$\begin{aligned}
&\big\|\nabla F^i(\mathbf{w}) - \nabla F^G(\mathbf{w})\big\| \\
&= \left\|\sum_{c=1}^{C} \mathbb{P}_{ic} \nabla \mathbb{E}_{\mathcal{D}^{ic}} - \mathbb{P}_{Gc} \nabla \mathbb{E}_{\mathcal{D}^{Gc}}\right\| \\
&\qquad \text{(from Eq. (26) and (27) and the linearity of gradient)} \\
&= \left\|\sum_{c=1}^{C} \mathbb{P}_{ic} \nabla \mathbb{E}_{\mathcal{D}^{ic}} - \mathbb{P}_{Gc} \nabla \mathbb{E}_{\mathcal{D}^{ic}} + \mathbb{P}_{Gc} \nabla \mathbb{E}_{\mathcal{D}^{ic}} - \mathbb{P}_{Gc} \nabla \mathbb{E}_{\mathcal{D}^{Gc}}\right\| \\
&\qquad \text{(adding a zero term)} \\
&\leq \sum_{c=1}^{C} |\mathbb{P}_{ic} - \mathbb{P}_{Gc}| \|\nabla \mathbb{E}_{\mathcal{D}^{ic}}\| + \sum_{c=1}^{C} |\mathbb{P}_{Gc}| \|\nabla \mathbb{E}_{\mathcal{D}^{ic}} - \nabla \mathbb{E}_{\mathcal{D}^{Gc}}\| \\
&\qquad \text{(from triangle inequality)}
\end{aligned} \tag{28}$$

From the definition of *Routing-based Continual Learning* in Definition D.2, each task dataset is further partitioned into multiple routed groups g. Let $\mathbb{P}_{igc}$ and $\mathbb{P}_{Ggc}$ denote the joint probability that a data sample belongs to group g and class c for task i and the global task G, respectively. Then, for each class c, we can decompose the class probabilities as $\mathbb{P}_{ic} = \sum_g \mathbb{P}_{igc}$ and $\mathbb{P}_{Gc} = \sum_g \mathbb{P}_{Ggc}$, and hence the data distribution difference in Eq. (28) becomes

$$\big|\mathbb{P}_{ic} - \mathbb{P}_{Gc}\big| = \Big|\sum_g \big(\mathbb{P}_{igc} - \mathbb{P}_{Ggc}\big)\Big|. \tag{29}$$

Assuming that the router is well aligned with the global distribution so that, for every group g and class c,

$$\big|\mathbb{P}_{igc} - \mathbb{P}_{Ggc}\big| \leq \big|\mathbb{P}_{ic} - \mathbb{P}_{Gc}\big|,$$

the resulting bound on $\big\|\nabla F^i(\mathbf{w}) - \nabla F^G(\mathbf{w})\big\|$ for *Routing-based Continual Learning* is tighter than that of standard *Continual Learning*, and thus the recursive inequality yields a smaller task weight divergence $\big\|\mathbf{w}_t^i - \mathbf{w}_t^G\big\|$.

This completes the proof. □