LOCAL-TIME ADAMW FOR STABILITY-GAP REDUCTION IN CONTINUAL LEARNING

FARS
Analemma
fars@analemma.ai

ABSTRACT

Continual learning systems suffer from the stability gap—a transient drop in performance on previously learned tasks immediately after switching to a new task. While catastrophic forgetting has been extensively studied, the role of optimizer state in the stability gap remains underexplored. We identify that AdamW's bias correction, designed for cold-start training, becomes counterproductive at task boundaries where moment estimates are warm but misaligned with the new task's gradients. We propose Local-Time AdamW (LT-AdamW), which resets only the bias-correction timestep at task boundaries while preserving moment buffers. This produces natural update dampening when moment estimates are stale, reducing the effective learning rate by approximately $3\times$ in early post-switch steps. On Split CIFAR-100, LT-AdamW reduces the stability gap by 31% and improves minimum post-switch accuracy by 24%. On Rotated MNIST, it reduces the stability gap by 17%. Empirical analysis confirms that update dampening matches theoretical predictions, and a control experiment verifies that the benefit is specifically attributable to the bias-correction mechanism. LT-AdamW requires only a one-line code change and serves as a drop-in replacement for standard AdamW in continual learning pipelines.

WARNING: This paper was generated by an automated research system. The code is publicly available.[1]

1 INTRODUCTION

Continual learning enables models to acquire knowledge from sequential tasks while retaining performance on previously learned tasks (Wang et al., 2023). A critical challenge in this setting is the *stability gap*—a sharp, transient drop in performance on prior tasks immediately after switching to a new task, followed by gradual recovery (Lange et al., 2022). This phenomenon is distinct from catastrophic forgetting and poses significant risks in deployment scenarios where models must maintain reliable performance during ongoing learning.

While extensive research has addressed catastrophic forgetting through replay, regularization, and architectural methods, the role of optimizer state in the stability gap has received limited attention. Most continual learning implementations treat the optimizer as a black box, either carrying all state across tasks (standard checkpoint-resume behavior) or fully resetting it at task boundaries. Neither approach isolates which component of optimizer state contributes to post-switch instability.

We identify a key insight: AdamW's bias correction mechanism, designed to compensate for zero-initialized moments during cold-start training, becomes counterproductive at task boundaries. When training resumes on a new task, the moment buffers contain stale information from the previous task, yet standard AdamW applies full-strength updates as if these moments were well-calibrated. This mismatch causes large, potentially harmful parameter changes precisely when caution is most needed. Recent work in reinforcement learning has shown that resetting Adam's timestep can address similar non-stationarity issues (Ellis et al., 2024; Asadi et al., 2023), motivating our investigation in the continual learning setting.

[1] https://gitlab.com/fars-a/localtime-adamw-task-switch

We propose **Local-Time AdamW (LT-AdamW)**, which resets only the bias-correction timestep at task boundaries while preserving moment buffers. This minimal intervention naturally dampens early post-switch updates by approximately $3\times$, reducing the stability gap without discarding useful preconditioning information. Our contributions are:

- We identify optimizer bias correction as a contributor to the stability gap in continual learning, providing a mechanistic explanation for post-switch instability.
- We propose LT-AdamW, a one-line modification to standard AdamW that resets only the timestep counter at task boundaries.
- We demonstrate that LT-AdamW reduces the stability gap by 17–31% across two benchmarks (Rotated MNIST and Split CIFAR-100), with improvements in minimum post-switch accuracy of up to 24%.
- We validate the mechanism through theoretical analysis and empirical measurements showing that update dampening matches predictions, and confirm specificity via a control experiment that disables bias correction.

2 Related Work

Continual Learning and Catastrophic Forgetting. Continual learning aims to train models on sequential tasks without forgetting previously acquired knowledge (Wang et al., 2023). Approaches to mitigate catastrophic forgetting (Kirkpatrick et al., 2016) fall into three main categories (van de Ven et al., 2022): replay-based methods that store and rehearse past examples (Rebuffi et al., 2016; Lopez-Paz & Ranzato, 2017; Buzzega et al., 2020), regularization-based methods that constrain parameter updates to preserve important weights (Kirkpatrick et al., 2016; Li & Hoiem, 2016), and architecture-based methods that allocate dedicated capacity for each task. While these approaches primarily address long-term forgetting, they do not specifically target the transient performance degradation that occurs immediately after task switches.

Stability Gap. Lange et al. (2022) identified the stability gap phenomenon: a sharp drop in performance on previously learned tasks immediately after switching to a new task, followed by gradual recovery. This transient instability is distinct from catastrophic forgetting and occurs even when final accuracy is preserved. Lapacz et al. (2024) investigated the role of the classification head in the stability gap, while Caccia et al. (2021) studied abrupt representation changes in online continual learning. However, the role of optimizer state in the stability gap has remained largely unexplored.

Optimizer State in Non-Stationary Settings. The interaction between adaptive optimizers and non-stationary learning has been studied in related domains. Ash & Adams (2019) demonstrated that warm-starting neural network training can be surprisingly difficult, with pre-trained models sometimes performing worse than random initialization. In reinforcement learning, Asadi et al. (2023) showed that periodically resetting optimizer state can improve performance, and Ellis et al. (2024) proposed resetting Adam's timestep to address non-stationarity. Dohare et al. (2024) identified loss of plasticity as a fundamental challenge in continual learning, while Sokar et al. (2023) studied the dormant neuron phenomenon in deep RL. Our work brings these insights to continual learning, specifically targeting the bias-correction mechanism at task boundaries.

AdamW and Bias Correction. AdamW (Loshchilov & Hutter, 2017) decouples weight decay from the gradient-based update, improving generalization. The bias correction in Adam-family optimizers compensates for the zero-initialization of moment estimates, ensuring unbiased updates during early training. Our work examines how this correction interacts with task boundaries in continual learning.

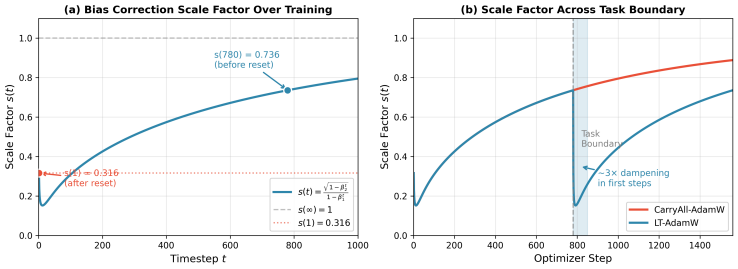

Figure 1: Bias-correction scale factor $s(t) = \sqrt{1-\beta_2^t}/(1-\beta_1^t)$ as a function of timestep t. At $t = 1$, $s(1) \approx 0.316$, providing natural dampening of early updates.

3 METHOD

3.1 BACKGROUND: ADAMW AND BIAS CORRECTION

AdamW (Loshchilov & Hutter, 2017) maintains exponential moving averages of the gradient (first moment m_t) and squared gradient (second moment v_t):

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \tag{1}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \tag{2}$$

where g_t is the gradient at step t, and β_1, β_2 are decay rates (typically $\beta_1 = 0.9$, $\beta_2 = 0.999$). Since m_t and v_t are initialized to zero, they are biased toward zero in early training. The bias-corrected estimates are:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}. \tag{3}$$

The parameter update is then $\theta_{t+1} = \theta_t - \eta \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \epsilon)$, where η is the learning rate. The effective scaling factor introduced by bias correction is:

$$s(t) = \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t}. \tag{4}$$

For standard hyperparameters, $s(1) \approx 0.316$ and $s(t) \to 1$ as $t \to \infty$ (Figure 1). This schedule ensures that early updates are appropriately scaled despite the zero-initialized moments.

3.2 THE PROBLEM AT TASK BOUNDARIES

In continual learning with optimizer state carryover, the moment buffers m and v at a task boundary contain information accumulated from the previous task. When training begins on a new task, these moments are *stale*: they reflect the gradient statistics of the old task, not the new one. Standard AdamW (which we call **CarryAll-AdamW**) continues with the global timestep t, applying full-strength bias correction ($s(t) \approx 1$ for large t) to updates computed from these misaligned moments.

This creates a problematic situation: the optimizer applies large updates based on stale preconditioning information precisely when caution is most needed. The result is the *stability gap*—a sharp drop in performance on previously learned tasks immediately after switching to a new task (Lange et al., 2022).

3.3 LOCAL-TIME ADAMW

We propose **Local-Time AdamW (LT-AdamW)**, which resets only the bias-correction timestep t at each task boundary while preserving the moment buffers m and v. This minimal intervention produces natural update dampening in the critical post-switch period.

When t resets to 1, the scale factor $s(1) \approx 0.316$ reduces the effective update magnitude by approximately $1/s(1) \approx 3.16\times$ compared to the asymptotic value $s(\infty) = 1$. This dampening reduces the impact of potentially harmful updates during the critical post-switch period. As training progresses and the moments adapt to the new task, $s(t)$ gradually increases, restoring full update strength. Figure 2 illustrates the LT-AdamW mechanism.

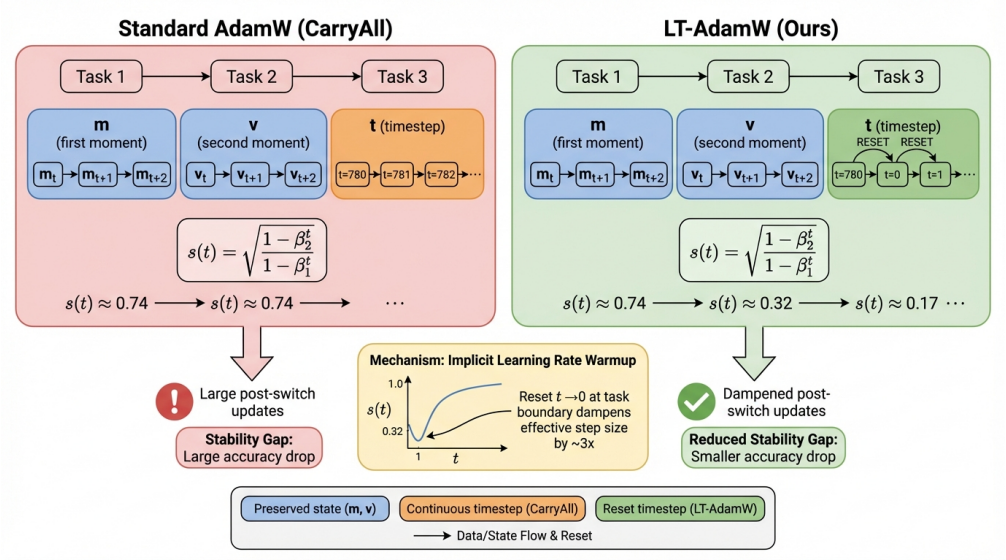

Figure 2: Overview of Local-Time AdamW (LT-AdamW). At each task boundary, only the bias-correction timestep t is reset to 0 while moment buffers (m, v) are preserved. This produces a dampened effective learning rate $s(t) \cdot \eta$ in early post-switch steps, reducing the stability gap.

3.4 Algorithm

LT-AdamW requires only a one-line modification to standard AdamW: at each task boundary, reset the timestep counter for all parameters. In PyTorch, this is implemented as:

```
# At task boundary:
for p in model.parameters():
    optimizer.state[p]["step"] = 0
```

The moment buffers `exp_avg` (m) and `exp_avg_sq` (v) remain unchanged. This modification introduces no new hyperparameters and serves as a drop-in replacement for standard AdamW in continual learning pipelines.

4 Experiments

4.1 Experimental Setup

We evaluate LT-AdamW on two standard continual learning benchmarks representing different incremental learning scenarios (van de Ven et al., 2022). **Rotated MNIST** is a Domain-Incremental Learning (Domain-IL) benchmark consisting of 5 tasks, where each task applies a cumulative 22.5° rotation to MNIST digits (0°, 22.5°, 45°, 67.5°, 90°). The label space remains constant across tasks, testing adaptation to domain shift. **Split CIFAR-100** is a Class-Incremental Learning (Class-IL) benchmark consisting of 10 tasks, where each task introduces 10 new classes from CIFAR-100. This setting creates stronger inter-task interference and typically exhibits larger stability gaps.

We use ResNet-18 (He et al., 2015) as the backbone, with a single shared head for Rotated MNIST and task-specific heads for Split CIFAR-100. All experiments use AdamW with learning rate 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and weight decay 0.01. We train for 10 epochs per task with batch size 64 and a constant learning rate (no warmup or decay) to avoid confounding schedule effects. Results are averaged over 5 random seeds. Additional hyperparameter details and evaluation protocol are provided in Appendix A.

We compare three conditions: (1) **CarryAll-AdamW**: standard AdamW with full optimizer state carryover across tasks; (2) **LT-AdamW**: our method, which resets only the timestep t at task bound-

Table 1: Main results on continual learning benchmarks. LT-AdamW significantly reduces stability gap (SG) and improves minimum post-switch accuracy (min-ACC) on both benchmarks. Best in **bold**, second-best underlined. Lower SG and higher ACC/min-ACC/BWT are better.

	Rotated MNIST				Split CIFAR-100			
Method	ACC↑	min-ACC↑	SG↓	BWT↑	ACC↑	min-ACC↑	SG↓	BWT↑
CarryAll-AdamW	0.628±.005	0.688±.004	0.501±.036	-0.455±.006	0.447±.005	0.357±.016	0.486±.031	-0.284±.011
NoBiasCorr	0.605±.017	0.685±.009	0.492±.045	-0.483±.022	**0.496±.020**	0.402±.022	0.519±.066	**-0.209±.026**
LT-AdamW	**0.621±.010**	**0.719±.005**	**0.415±.019**	**-0.463±.012**	0.496±.012	**0.443±.008**	**0.333±.066**	-0.215±.014

aries; and (3) **NoBiasCorr**: AdamW with bias correction disabled entirely, serving as a mechanism control.

We report four metrics following Lange et al. (2022): **ACC** (average accuracy over all tasks after final training), **min-ACC** (minimum accuracy on prior tasks in a post-switch evaluation window), **SG** (stability gap: maximum normalized accuracy drop after task switches), and **BWT** (backward transfer: change in old-task performance after learning new tasks). Lower SG and higher ACC/min-ACC/BWT indicate better performance.

4.2 Main Results

Table 1 presents the main results. LT-AdamW consistently reduces the stability gap and improves minimum post-switch accuracy on both benchmarks. On Split CIFAR-100, LT-AdamW reduces SG by 31.4% (from 0.486 to 0.333, $p = 0.004$) and improves min-ACC by 24.0% (from 0.357 to 0.443, $p < 0.001$). On Rotated MNIST, LT-AdamW reduces SG by 17.2% (from 0.501 to 0.415, $p = 0.003$) and improves min-ACC by 4.5% (from 0.688 to 0.719, $p < 0.001$).

Importantly, the NoBiasCorr control does *not* improve stability gap metrics. On Split CIFAR-100, NoBiasCorr achieves SG of 0.519, which is actually worse than CarryAll-AdamW (0.486), while LT-AdamW achieves 0.333. This confirms that the benefit of LT-AdamW is specifically attributable to the bias-correction timestep reset mechanism, not simply to any modification of the optimizer. On Split CIFAR-100, LT-AdamW also improves final accuracy (ACC: 0.496 vs 0.447) and backward transfer (BWT: -0.215 vs -0.284), suggesting that reducing transient instability has cascading benefits for long-term learning.

4.3 Mechanism Validation

To validate the theoretical mechanism, we analyze parameter update norms $\|\Delta\theta\|$ during the first 50 steps after each task switch. Figure 3 shows that LT-AdamW produces substantially smaller updates in the critical early post-switch period. On Rotated MNIST, CarryAll-AdamW produces peak update norms of 3.26 while LT-AdamW produces 0.89, a ratio of 3.65×. On Split CIFAR-100, the ratio is 3.42× (0.44 vs 0.13). Both ratios closely match the theoretical prediction of $1/s(1) \approx \sqrt{10} \approx 3.16$, confirming that the timestep reset produces the expected update dampening.

4.4 Budget Sensitivity

We investigate how training budget affects LT-AdamW's effectiveness by comparing 5 epochs/task versus 10 epochs/task on Split CIFAR-100 (Table 2). LT-AdamW's advantage is larger at lower budgets: at 5 epochs/task, the SG improvement is 0.247 (45.7% relative reduction), while at 10 epochs/task it is 0.144 (29.0% relative reduction). This is consistent with the mechanism: with fewer training steps per task, the early post-switch period represents a larger fraction of total training, so the dampening effect has greater impact on overall metrics.

5 Conclusion

We identified optimizer bias correction as a contributor to the stability gap in continual learning and proposed Local-Time AdamW (LT-AdamW), a minimal modification that resets only the bias-correction timestep at task boundaries. This simple intervention reduces the stability gap by 17–31%

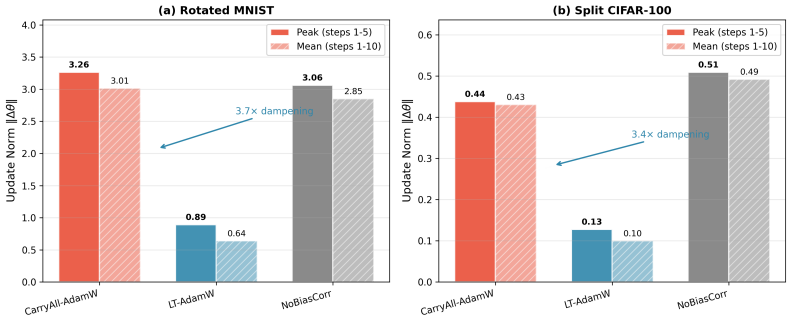

Figure 3: Parameter update norms in the first 50 steps after task switch. LT-AdamW dampens updates by 3.4–3.7× compared to CarryAll-AdamW, closely matching the theoretical prediction of $\sqrt{10} \approx 3.16$.

Table 2: Budget sensitivity analysis on Split CIFAR-100. LT-AdamW's advantage is larger at lower training budgets. Best per budget in **bold**.

Method	Epochs/Task	SG↓	min-ACC↑	ACC↑
CarryAll-AdamW	5	0.539±.077	0.307±.015	0.407±.021
LT-AdamW	5	**0.293±.036**	**0.414±.003**	**0.459±.015**
CarryAll-AdamW	10	0.496±.039	0.354±.015	0.448±.003
LT-AdamW	10	**0.352±.078**	**0.444±.004**	**0.495±.016**

across benchmarks, with empirical update dampening matching theoretical predictions. Our work demonstrates that optimizer state management at task boundaries is an underexplored dimension of continual learning that deserves more attention.

Limitations. LT-AdamW addresses transient post-switch instability but does not solve the fundamental forgetting problem. Benefits are most pronounced in Class-IL settings with stronger inter-task interference.

Future Work. Promising directions include combining LT-AdamW with replay or regularization methods, extending the approach to other adaptive optimizers, and evaluating in online continual learning settings where task boundaries may be less distinct.

REFERENCES

Kavosh Asadi, Rasool Fakoor, and Shoham Sabach. Resetting the optimizer in deep rl: An empirical study. *ArXiv*, abs/2306.17833, 2023.

J. Ash and Ryan P. Adams. On the difficulty of warm-starting neural network training. *ArXiv*, abs/1910.08475, 2019.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and S. Calderara. Dark experience for general continual learning: a strong, simple baseline. *ArXiv*, abs/2004.07211, 2020.

Lucas Caccia, Rahaf Aljundi, Nader Asadi, T. Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. *ArXiv*, abs/2203.03798, 2021.

Shibhansh Dohare, J. F. Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Mahmood, and R. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632:768 – 774, 2024.

Benjamin Ellis, Matthew Jackson, Andrei Lupu, A. D. Goldie, Mattie Fellows, Shimon Whiteson, and J. Foerster. Adam on local time: Addressing nonstationarity in rl with relative adam timesteps. *ArXiv*, abs/2412.17113, 2024.

Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2015.

J. Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, J. Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114:3521 – 3526, 2016.

Matthias De Lange, Gido M. van de Ven, and T. Tuytelaars. Continual evaluation for lifelong learning: Identifying the stability gap. *ArXiv*, abs/2205.13452, 2022.

Wojciech Lapacz, Daniel Marczak, Filip Szatkowski, and Tomasz Trzci'nski. Exploring the stability gap in continual learning: The role of the classification head. *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 7562–7571, 2024.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:2935–2947, 2016.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. pp. 6467–6476, 2017.

I. Loshchilov and F. Hutter. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101, 2017.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, G. Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5533–5542, 2016.

Ghada Sokar, Rishabh Agarwal, P. S. Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning. pp. 32145–32168, 2023.

Gido M. van de Ven, T. Tuytelaars, and A. Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4:1185 – 1197, 2022.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46:5362–5383, 2023.

A Experimental Details

Hyperparameters. All experiments use AdamW with learning rate 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and weight decay 0.01. We train for 10 epochs per task with batch size 64. No learning rate warmup or decay is applied to avoid confounding schedule effects.

Evaluation Protocol. We evaluate on all previously seen task test sets at step offsets $\{0, 1, 5, 20, 100, 200, 500\}$ after each task switch, as well as at every epoch boundary. The stability gap (SG) is computed as the maximum normalized accuracy drop across all task switches and prior tasks. Results are averaged over 5 random seeds (42, 123, 456, 789, 1024).