HiLD 2025: 3rd Workshop on High-dimensional Learning Dynamics 1–15

Pathwise EMA: An Intrinsic Clock for Weight Averaging

author names withheld

Under Review for the Workshop on High-dimensional Learning Dynamics, 2025

Abstract

Exponential moving averages of model weights are widely used to stabilize deep learning training, but standard EMA introduces decay, offset, and update-frequency hyperparameters that must be retuned across learning-rate schedules, batch sizes, and model scales. We ask whether EMA can instead be made adaptive to the optimization trajectory itself. We propose Pathwise EMA (PEMA), a parameter-free EMA scheme that replaces time-based decay with a decay rule based on the normalized path length traveled by the model parameters in weight space. The central intuition is that path length acts as an intrinsic clock for training: high-velocity or noisy trajectories require stronger smoothing, whereas slower trajectories require less smoothing to avoid lag. Across supervised fine-tuning experiments on SmolLM2, Qwen, and Gemma models, PEMA consistently matches or outperforms the best tuned standard EMA across sweeps over learning rate, minimum learning rate, batch size, and update frequency. These results suggest that path-based averaging can provide a simple, robust stabilizer for language model fine-tuning while substantially reducing the hyperparameter tuning burden of EMA.

1. Introduction

Motivated by the theory of stochastic optimization (Polyak and Juditsky, 1992; Ruppert, 1988), taking an exponential moving average (EMA) of model parameters throughout training has become increasingly ubiquitous in deep learning (Block and Zhang, 2025; Block et al., 2024; Izmailov et al., 2018; Kaddour, 2022; Busbridge et al., 2023). Indeed, most modern open-source language models incorporate some form of iterate averaging into their training pipelines (Liu et al., 2024; OLMo et al., 2024; Lambert et al., 2024; Grattafiori et al., 2024). While the practical benefits of EMA are well-known, one major barrier its further spread is the fact that it introduces several new hyperparameters, whose tuning leads to additional cost. We thus ask: *Is it possible to instantiate an EMA scheme that is robust to hyperparameter choice?*

To be more precise, when training a model with parameters θ, a standard EMA update takes the form

$$\theta_{t+1}^{\text{EMA}} = (1 - \beta_t)\theta_t^{\text{EMA}} + \beta_t \theta_{t+1} \tag{1}$$

where β_t is a decay factor that typically follows a schedule of the form $\beta_t = (t + \tau)^{-\gamma}$, with τ and γ being hyperparameters that must be tuned for each training run and θ_{t+1} being the model parameters after the $(t+1)^{\text{st}}$ optimizer update . Moreover, in practice, due to a desire to reduce training time, there is a further hyperparameter choice of update frequency, i.e. how often to perform the EMA update. Motivated by the theory of stochastic

processes (Le Gall, 2016), in this work we propose a novel parameter-free EMA (PEMA) scheme that changes the decay factor β_t to be a function of the *normalized path length* of the model parameters, which is a measure of how far the model has traveled in weight space. We show that this formulation allows PEMA to adapt to the optimization trajectory of a training run, and thus be robust to hyperparameter choice across a wide range of training configurations. We validate our method on language modeling tasks, showing that it can match or outperform standard EMA with tuned hyperparameters, while being robust to hyperparameter choice across different learning rate schedules and model scales.

Section 3 presents our core experimental results on small and large-scale language models. We provide comprehensive supporting evidence in the appendices. Appendix C contains exhaustive hyperparameter sweeps for models up to 1B parameters, as well as stress tests in data-scarce (Section C.3) and non-monotonic optimization regimes. Finally, we include an ablation study comparing global versus per-parameter scaling and provide full pseudocode for our implementation (Algorithm 1) to ensure reproducibility.

2. Deriving Parameter-Free EMA

As we clarified above, the standard EMA update rule from (1) requires tuning a number of hyperparameters, most critically the power γ. We propose modifying this update to instill adaptivity to the optimization trajectory by relying on *normalized path length* of the training trajectory as opposed to the step count t. In particular, the theory of stochastic processes suggests that *quadratic variation* of the optimization trajectory is a natural intrinsic time variable for stochastic optimization (Le Gall, 2016); in our experiments, however, we find that *total variation* (i.e. path length) is a more effective measure of intrinsic time for EMA, providing preliminary evidence that, at least in the finetuning settings we consider below, the optimization process is not noise dominated.

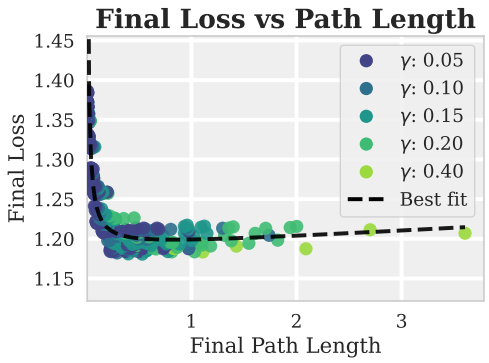

Figure 1: Best loss for all sweeps of SmolLM2-135M achieved by standard EMA relative to the total path length $\ell(T)$. Each dot represents a training configuration, and the color indicates the best EMA power.

As shown in Algorithm 1, we first calculate the standard deviation for each parameter tensor at initialization: $\sigma_p = \mathrm{std}(\theta_{0,p}) + \epsilon$. We then define the cumulative normalized path length $\ell(t)$ as the average movement across all parameters N:

$$\ell(t) = \frac{1}{N} \sum_{p \in \text{Params}} \sum_{k=1}^{t/f} \frac{\|\theta_{k \cdot f,p} - \theta_{(k-1) \cdot f,p}\|_1}{\sigma_p} \tag{2}$$

where f is the update frequency. This metric tracks the total distance traveled by the weights in a normalized coordinate system. We then reformulate the EMA coefficient β_t used in Equation 1 as:

$$\beta_t = (1 + T \cdot \ell(T) \cdot \ell(t))^{-\gamma} \tag{3}$$

This implementation was driven by empirical observations during our finetuning of SmolLM2-135M (Allal et al., 2025). As illustrated in Figure 1, we analyzed multiple training configurations. We observed that a total path length $\ell(T)$ of approximately 0.5 appears to be optimal for standard EMA at a specific power. Path lengths that are significantly higher or lower than this threshold consistently lead to sub-optimal results across various fixed EMA schedules.

Crucially, the data reveal that the best-performing EMA power is not constant: higher path lengths (noisier or more aggressive training) require higher EMA powers to stabilize the weights, whereas lower path lengths (stable or decaying learning rates) require lower powers to avoid lagging. We took a reference path length of 1.0 as a calibration point, noting that a power of $\gamma \approx 0.15$ performs best in this regime. In such a scenario, our formulation for β_t approaches the standard time-based EMA values.

We hypothesize that this optimal path length relationship scales predictably with model size when training on the same or similar data distributions. While a significant shift in data distribution or task complexity might alter the underlying optimization manifold, our tests, wherein scaling from 135M to 1B parameters on similar datasets (see Section 3.1), indicate that this curve remains stable.

To make PEMA adaptive, we introduced the term $T \cdot \ell(T) \cdot \ell(t)$ in Equation (3) to replace the standard time variable. This choice was motivated by the need to match curves across different optimization schedules. In experiments with LR decay, the path length $\ell(t)$ starts steep and gradually levels off as the learning rate diminishes. By using $\ell(t)$ in the denominator, β_t automatically captures this non-linearity: the EMA maintains lower inertia early in training when the model is traversing the weight space quickly, but increases inertia more aggressively as the model converges, and the path length flattens.

Furthermore, this formulation is related to the standard EMA formula, which was tested on SmolLM2-135M, and later validated on the 360M model. We observed that $\ell(t)$ behaved roughly as a straight line from 0 to 1 (Figure 3) in configurations with a high learning rate and no LR decay. In such cases, the term $T \cdot \ell(T) \cdot \ell(t)$ simplifies to approximately $T \cdot 1 \cdot \frac{t}{T} = t$. Consequently, our formulation for β_t approaches the same numerical values as the standard time-based EMA under these conditions. Since a power of $\gamma = 0.15$ performed optimally for the 135M model in this setting, the implementation detailed in Algorithm **??** allows the weight updates to remain robust under different training configurations.

3. Experiments

To test the efficacy of PEMA, we evaluate our approach across various training configurations and model architectures. More specifically, we conduct sweeps on SmolLM2-135M and SmolLM2-360M using the Smol-smoltalk dataset (Allal et al., 2025), and on gemma-3-1b-pt (Gemma Team et al., 2025) and Qwen3-0.6B-Base (Qwen Team, 2025) using the Tulu-3-sft dataset (Lambert et al., 2024). Additional details on the training configurations are provided in Table 1 in the appendix. We first examine the geometric properties of the parameter updates through weight path analysis, followed by comprehensive ablation studies and stress tests in data-scarce and non-monotonic optimization regimes.

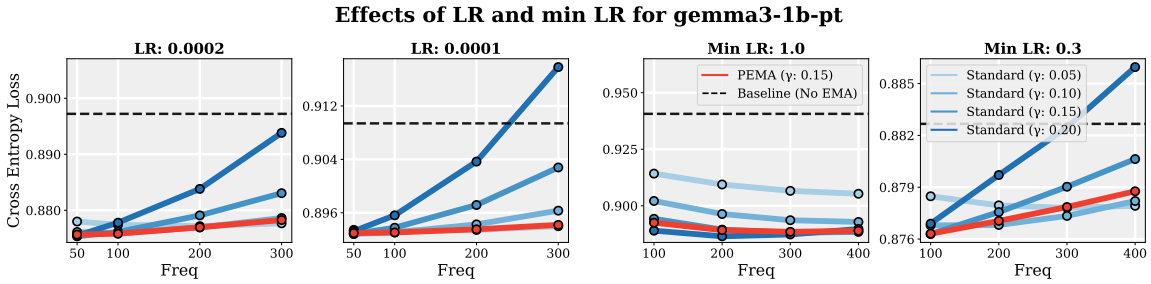

Figure 2: Learning Rate (LR) and Minimum LR sweeps for gemma-3-1B-pt on the Tulu-3-SFT dataset. Standard EMA configurations exhibit performance trade-offs: low powers (0.05-0.10) struggle with low update frequencies and low decay, whereas high powers (0.15-0.20) degrade when the path traveled is low. **PEMA (red line) consistently matches the best-performing standard EMA.**

3.1. Results on Large Models

To validate the scalability of our approach, we evaluated PEMA on gemma-3-1B-pt. As shown in Figure 2, we conducted sweeps over the peak learning rate and minimum learning rate using the Tulu-3-SFT dataset. The results highlight a limitation of standard EMA: its high sensitivity to specific optimization trajectories. For standard EMA configurations, low powers (e.g., 0.05 and 0.10) result in degraded performance when update frequencies are low or when the learning rate decay is minimal. Conversely, higher powers (e.g., 0.15 and 0.20) perform poorly in regimes with high update frequencies and high learning rate decay—scenarios where the total path length traveled by the weights is relatively small. In these cases, high-power standard EMA lags too far behind the current parameters, failing to capture the most recent optimization gains. In contrast, our proposed PEMA ($\gamma = 0.15$) maintains consistent, high performance across all tested configurations.

3.2. Ablation Studies on Small Models

To demonstrate the robustness of our proposed scaling rule, we conducted ablation studies on SmolLM2-360M using the Smol-smoltalk dataset. These experiments investigate the model's ability to adapt to diverse training configurations. We fine-tuned the model using the default parameters listed in Table 1 and evaluated performance using 1,024 randomly sampled held-out inputs.

3.2.1. Effects of LR, LR Decay, and Batch size

PEMA demonstrated high robustness by varying peak learning rate, as shown in Figure 3. We varied the learning rate from 2×10^{-3}, which we found to be the highest stable learning rate for SmolLM-360M, to 2e-4 and the update frequency from 50 to 300. With a higher learning rate and lower update frequencies, the weight updates become noisier and larger, as shown in Figure 13. Under these conditions, we observe that more aggressive standard EMA powers of 0.2 and 0.15 offer better performance.

However, with a smaller learning rate and high update frequencies, the weight updates become slower, and aggressive EMA powers such as 0.2 and 0.15 start to lag. Similarly,

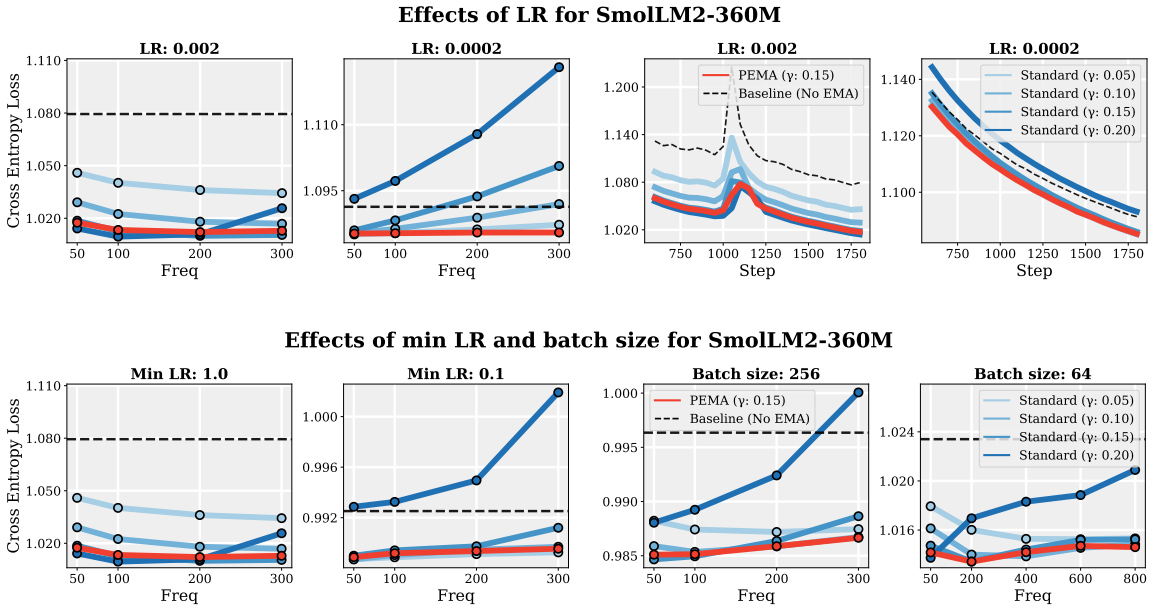

Figure 3: Ablation summary of PEMA vs. standard EMA on SmolLM2-360M. (Top) Varying peak learning rates and (Bottom) varying minimum learning rate and batch size: **PEMA remains competitive across all settings**, whereas standard EMA performance degrades under certain configurations.

the standard method with a low EMA power, such as 0.05, shows performance degradation for high learning rates, since it places more weight on the most recent model parameters, which are more likely to be noisy. However, our proposed method, with a power of 0.15, was able to adapt to both learning rates.

We also observed that PEMA is robust to varying learning rate decay and batch size, although the impact of these two hyperparameters overall was not as strong on standard EMA as the peak learning rate and update frequency. Having no learning rate decay, similar to having a high learning rate, also leads to large model weight updates that may be noisier. Because of this, PEMA was able to make slower weight updates by lowering β_t due to the larger path length. As a result, PEMA maintains high performance, while the standard method with an EMA power of 0.05 suffers due to a high β_t.

Finally, the effect of the batch size appeared to be the smallest, although larger EMA power shows slightly better performance for smaller batch sizes, which is expected due to noisier weight updates associated with a small batch size. And once again, PEMA was able to adapt by lowering β_t for small batch sizes, as shown in Figure 13, and maintain high performance across all configurations.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín,

Vaibhav Srivastav, et al. Smollm2: When smol goes big – data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*, 2025.

Adam Block and Cyril Zhang. Ema without the lag: Bias-corrected iterate averaging schemes. *arXiv preprint arXiv:2508.00180*, 2025.

Adam Block, Dylan J Foster, Akshay Krishnamurthy, Max Simchowitz, and Cyril Zhang. Butterfly effects of sgd noise: Error amplification in behavior cloning and autoregression. In *The Twelfth International Conference on Learning Representations*, 2024.

Dan Busbridge, Jason Ramapuram, Pierre Ablin, Tatiana Likhomanenko, Eeshan Gunesh Dhekane, Xavier Suau Cuadros, and Russell Webb. How to scale your ema. *Advances in Neural Information Processing Systems*, 36:73122–73174, 2023.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Jean Kaddour. Stop wasting my time! saving days of imagenet and bert training with latest weight averaging. *arXiv preprint arXiv:2209.14981*, 2022.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Jean-François Le Gall. *Brownian motion, martingales, and stochastic calculus*, volume 274. Springer, 2016.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

David Ruppert. Efficient estimations from a slowly convergent robbins-monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.

Algorithm 1 PEMA

Input: Model weights θ_t, power γ, update frequency f.
Set $\sigma_i \leftarrow \text{std}(\theta_{0,i})$ for all parameters i.
for *each checkpoint* t **do**
  if $t \bmod f = 0$ **then**
    $\ell_i(t) \leftarrow \ell_i(t-f) + \frac{1}{N}\sum_i \frac{1}{\sigma_i}|\theta_{t,i} - \theta_{t-f,i}|$ (%% Phase 1: Accumulate path length)
  end
end
Set $\theta_0^{EMA} \leftarrow \theta_0$.
for *each checkpoint* t **do**
  if $t \bmod f = 0$ **then**
    $\beta_t \leftarrow (1 + T \cdot \ell(T) \cdot \ell(t))^{-\gamma}$ (%% Phase 2: Compute weighted average)
    $\theta_{t+1}^{EMA} \leftarrow (1-\beta_t)\theta_t^{EMA} + \beta_t \theta_{t+1}$
  end
end
Return θ_{T+1}^{EMA}.

Appendix A. Discussion

Our empirical results demonstrate that Pathwise-EMA (PEMA) provides a robust, tuning-free alternative to standard Exponential Moving Average schedules, autonomously adapting to diverse training dynamics. By anchoring the decay factor to the normalized path length of the model weights, PEMA successfully mitigates the need for granular hyperparameter optimization across varying learning rates, update frequencies, and batch sizes.

However, it is important to contextualize the magnitude of these performance gains. While PEMA consistently matches or slightly outperforms the best-tuned standard EMA configurations across our sweeps, the absolute effect on the final objective is relatively modest, typically yielding approximately a 2% lower cross-entropy loss compared to baselines. This indicates that while PEMA is a highly reliable stabilizer and statistical regularizer, EMA serves primarily to refine and smooth the final optimization trajectory rather than act as a primary driver of model capability.

Appendix B. Future Work

A key limitation of our current study is the scope of the evaluated data distributions. Our ablation studies and large-model experiments primarily focused on supervised fine-tuning (SFT) tasks where the target data distribution aligns relatively well with the base models' original pretraining distributions. We have yet to conduct extensive testing across a much wider array of highly divergent data distributions.

We hypothesize that the optimal total path length, $\ell(T)$, is fundamentally tied to the distributional distance between the pretraining corpus and the fine-tuning dataset. In scenarios where a model is fine-tuned on a dataset that is significantly out-of-domain compared to its pretraining, the parameters must traverse a substantially larger distance in the weight space to capture the novel representations. In such cases, the empirical optimal path length curve we observed (where $\ell(T) \approx 0.5$ served as a strong anchor) is expected to shift higher.

Future work will focus on accounting for more extreme optimization landscapes and Reinforcement Learning, where weight updates are notoriously noisy. Additionally, investigating whether the path length scaling rule can be dynamically adjusted based on an online approximation of distributional shift could pave the way for a universally optimal, zero-shot EMA algorithm that requires absolute zero calibration regardless of the target domain.

Appendix C. Additional Ablation Results

In this section, we provide a comprehensive breakdown of the empirical results that informed the development of Pathwise-EMA (PEMA). We evaluate PEMA across a wide range of architectures (from 135M to 1B parameters) and stress-test the algorithm under non-monotonic optimization schedules and data-scarce regimes.

Param	SmolLM2-135M	SmolLM2-360M	Qwen3-0.6B-Base	gemma-3-1b-pt
Batch size	256	256	512	512
LR	2×10^{-3}	2×10^{-3}	2×10^{-4}	2×10^{-4}
Min LR	1	1	0.3	0.3
Warm up	100	100	100	100
LR scheduler	Linear	Linear	Linear	Linear
EMA power (γ)	0.15	0.15	0.15	0.15
Update freq	50	50	100	100
Training steps	1800	1800	1800	1800
Dataset	Smol-smoltalk	Smol-smoltalk	Tulu-3	Tulu-3
Eval size	1024	1024	4096	4096
Train GPU hours	5	12	24	36
Eval GPU hours	0.25	0.3	1	2

Table 1: Default configurations used for ablation experiments across all evaluated models. The training setup is a single RTX6000-Pro.

C.1. Ablation on large models

To ensure scaling laws translate to larger architectures, we performed exhaustive sweeps on gemma-3-1b-pt and Qwen3-0.6B-Base. Standard EMA performance proved highly sensitive to the interaction between learning rate, batch size, and update frequency.

C.2. Ablation on small models

The SmolLM2-135M and 360M models served as our primary testbeds for identifying the tuning-free properties of path-based decay. We specifically utilized the 135M model as our initial reference point for all calibration experiments before verifying the results on larger scales. Due to the low computational overhead of these smaller architectures—with each individual training run requiring fewer than 12 GPU hours—we were able to sweep through a significantly larger volume of hyperparameter configurations compared to our 1B+ parameter models. This high-throughput testing was instrumental in mapping the relationship between weight velocity and optimal EMA smoothing.

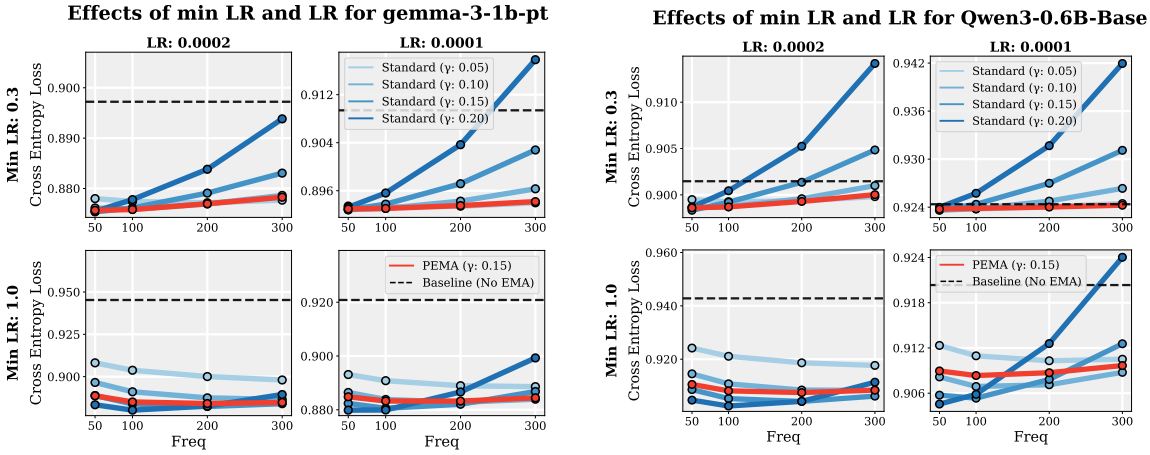

Figure 4: Sensitivity to learning rate (LR) and minimum LR for gemma-3-1b-pt and Qwen3-0.6B-Base. In fast parameter update regimes (high LR), low EMA powers struggle to regularize noise, while in slow parameter update regimes, high powers degrade performance by lagging. PEMA consistently matches the best-performing standard EMA by adaptively responding to these dynamics.

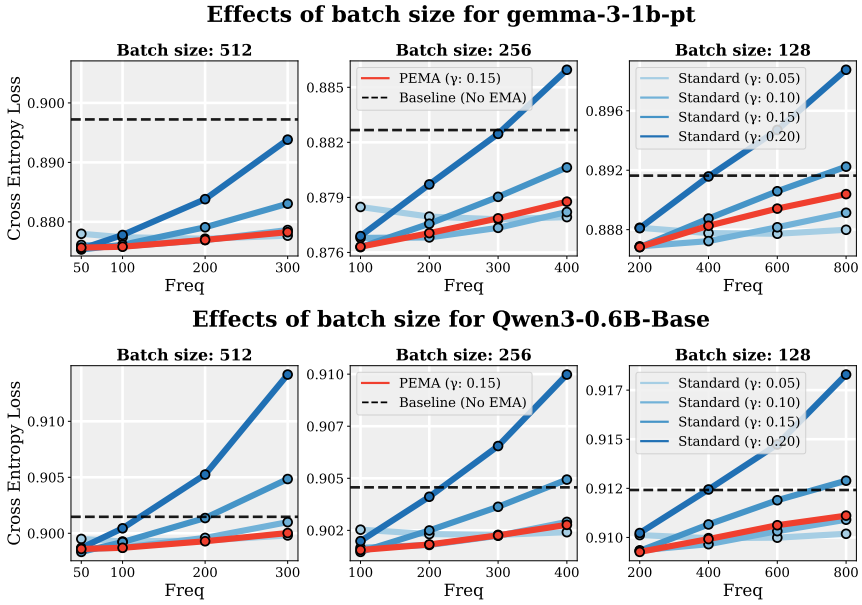

Figure 5: Batch size comparison for Gemma and Qwen models. While standard EMA ($\gamma = 0.20$) suffers steep penalties as update frequency increases due to gradient noise, PEMA maintains stability across batch sizes (128–512) by modulating smoothing based on observed weight movement.

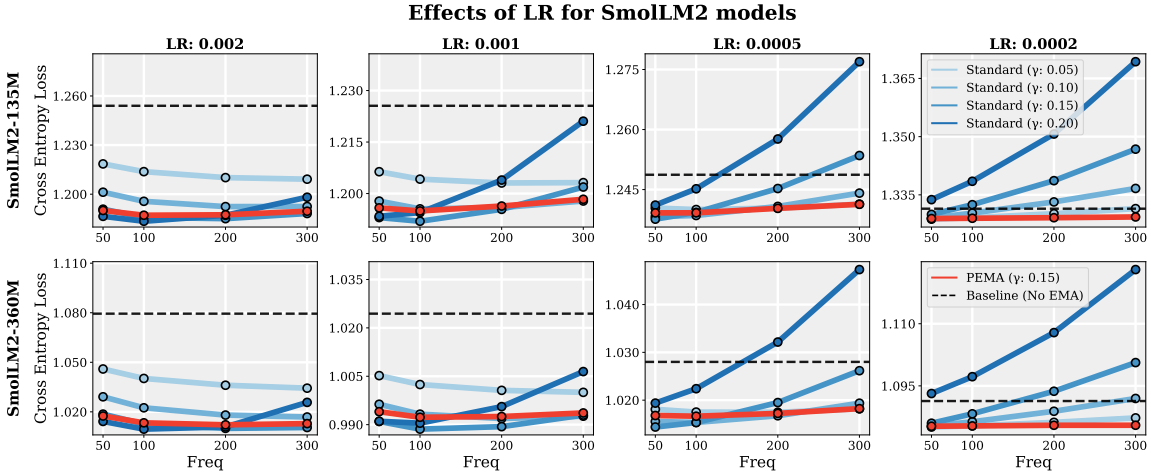

Figure 6: Learning rate sweep for SmolLM2-360M and SmolLM2-135M. At high learning rates (0.002), weight updates are naturally larger and noisier, leading to higher cumulative path lengths. Under these conditions, PEMA ($\gamma = 0.15$) consistently dampens the noise of large parameter updates, matching the optimal baseline across the entire spectrum and resolving the noise-sensitivity that causes low-power standard EMA (0.05) to degrade.

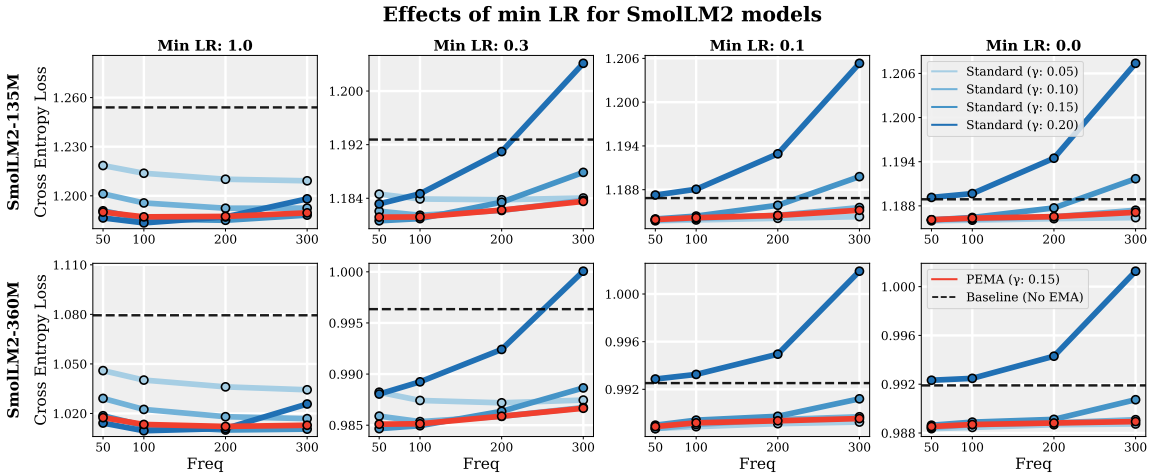

Figure 7: Minimum learning rate sweep for SmolLM2-360M and SmolLM2-135M. Standard EMA with $\gamma = 0.05$ is particularly vulnerable to configurations with no LR decay (Min LR: 1.0) because it places too much weight on the most recent updates in high-velocity regimes. PEMA robustly accounts for this movement by automatically increasing the smoothing factor based on the observed path length.

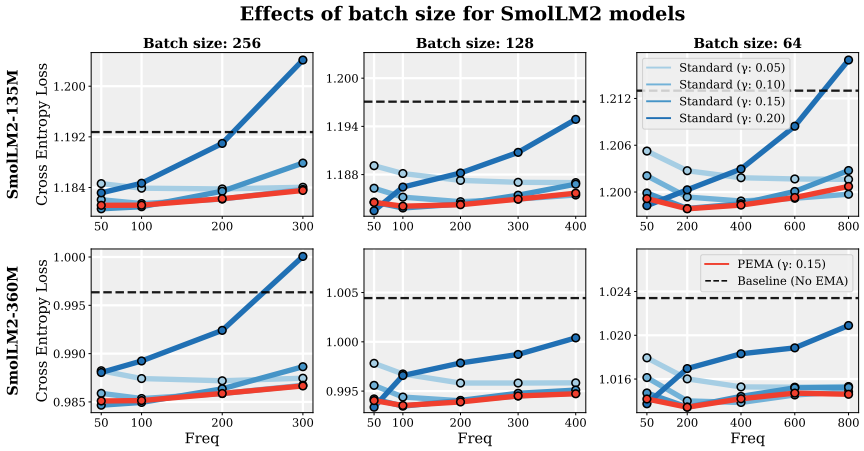

Figure 8: Batch size sweep for SmolLM2-360M and SmolLM2-135M. When batch sizes are reduced to 64, the optimization trajectory becomes increasingly erratic, causing standard high-power EMA ($\gamma = 0.20$) to frequently diverge at high update frequencies. PEMA prevents this divergence by linking the smoothing factor to the physical distance traveled in weight space, ensuring stability despite small-batch noise.

C.3. Effect on Statistical Regularization

To evaluate the effectiveness of our approach as a statistical regularizer, we tested the algorithm in a data-scarce regime by performing SFT for 4 epochs on a 25% subset of the Smol-smoltalk dataset. This setup is specifically designed to induce overfitting, allowing us to observe how different EMA strategies handle the transition from learning general patterns to memorizing noise.

As shown in Figures 9 and 10, standard EMA configurations exhibit high sensitivity to hyperparameters in this multi-epoch setting. In Figure 9, which compares two distinct learning rate regimes, we observe that while fine-tuning at a lower learning rate shows minimal overfitting, the higher learning rate induces significant loss spikes at the beginning of each epoch—particularly after the third—leading to severe performance degradation. Similarly, Figure 10 illustrates that a lack of learning rate decay exacerbates overfitting post-epoch three. While aggressive learning rate decay partially mitigates this, signs of memorization remain evident across standard baselines.

PEMA continues to account for variations in peak and minimum learning rates even in these overfitting-prone settings using only a single hyperparameter. However, we observe that a higher power ($\gamma = 0.3$) is required to maintain optimal performance compared to the 0.15 used in previous sections. This is due to a fundamental property of the scaling rule: the path length is a function of the training parameters—such as learning rate, batch size, and update frequency—rather than the underlying data distribution or the onset of overfitting. Because the optimization trajectory's physical distance does not inherently shorten when the model begins to overfit, a more aggressive power is necessary to provide the stronger statistical regularization required to dampen the noise associated with data scarcity. Despite this shift in the optimal power constant, PEMA remains robust, matching

the minimum loss of the best-performing tuned standard baselines across the tested learning rates.

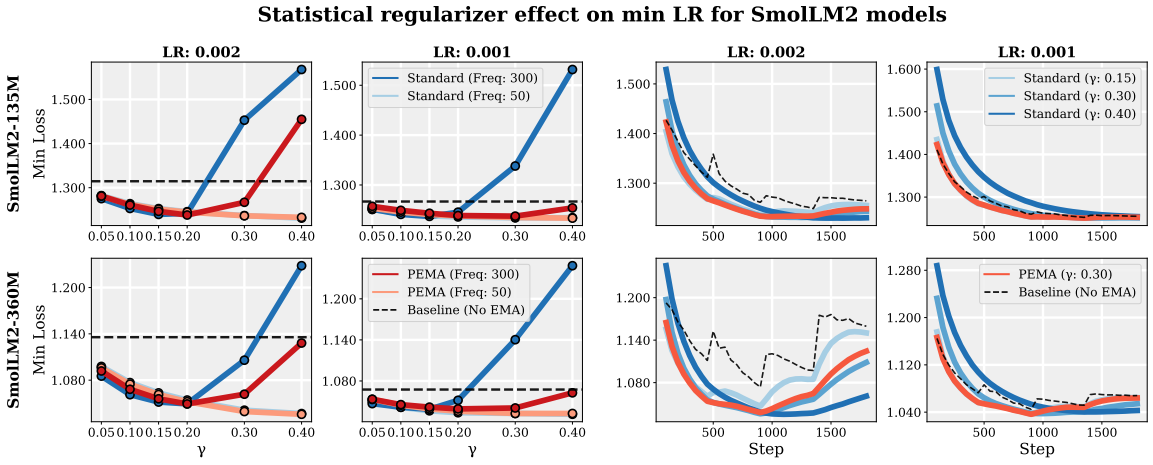

Figure 9: Learning rate sweep on a 25% subset of Smol-smoltalk over four epochs. In this data-scarce regime, standard methods are highly prone to overfitting. PEMA mitigates this by providing consistent, adaptive statistical regularization across the entire training trajectory while matching the performance of the best standard EMA model.

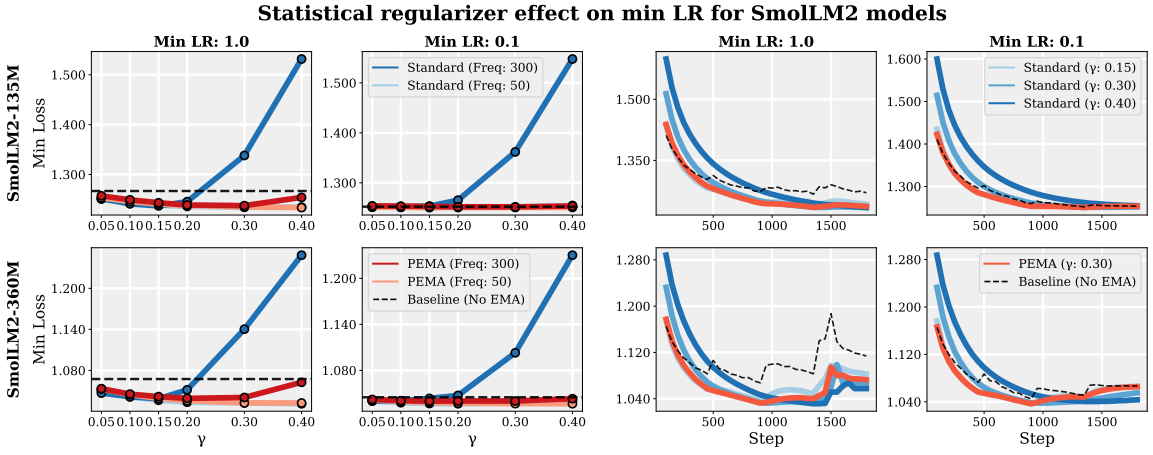

Figure 10: Minimum learning rate multiplier sweep on a 25% subset of Smol-smoltalk. PEMA adaptively adjusts for the increased path length caused by a sustained high learning rate. A base learning rate of 1×10^{-3} was used for this sweep, as 2×10^{-3} proved unstable without learning rate decay.

C.4. Cyclical Learning Rate Schedulers

To evaluate robustness, we tested a *Cosine Annealing with Restarts* scheduler (8 cycles) with peak LRs of 2×10^{-3} and 2×10^{-4} (Figure 11). Abrupt resets to peak LR induce frequent

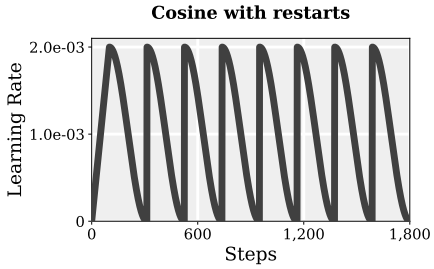

Figure 11: The cosine annealing with restarts scheduler with a peak learning rate of 2×10^{-3} and 8 cycles. This non-monotonic schedule tests optimization robustness, as abrupt resets to the peak learning rate induce frequent spikes and rapid changes in parameter velocity.

spikes in parameter velocity. As shown in Figure 12, our model maintains competitive performance at 2×10^{-3} and achieves optimal performance at 2×10^{-4} compared to standard methods. This confirms PEMA effectively handles non-monotonic schedules and recovers quickly from periodic instability.

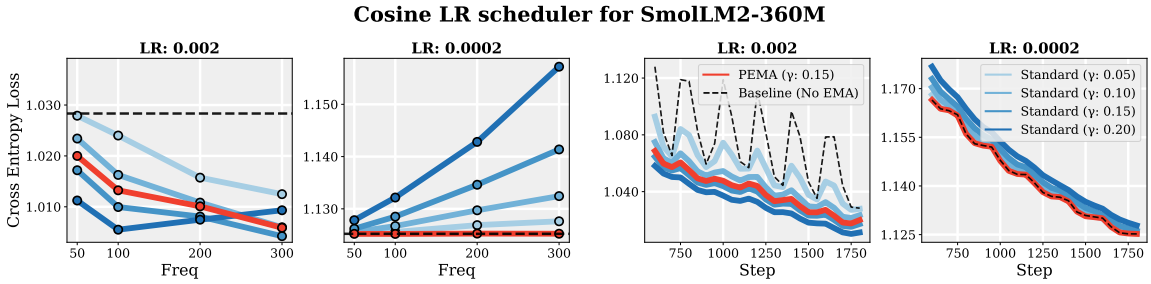

Figure 12: Cosine annealing with restarts on SmolLM2-360M. PEMA effectively handles non-monotonic schedules and recovers quickly from periodic instability. It maintains competitive performance at a high peak LR (2×10^{-3}) and achieves optimal performance at a lower peak LR (2×10^{-4}) compared to tuned standard methods.

C.5. Path of weights

We first examine the average path traveled by the parameters of the model at step t, defined in Equation 2, as well as the behavior of the EMA coefficient β_t. We observe these metrics over the course of SFT on SmolLM2-360M as illustrated in Figure 13.

The path length is highly sensitive to the chosen hyperparameters; smaller batch sizes and higher learning rates significantly increase the total distance parameters travel. Standard EMA applies a fixed decay schedule that cannot adapt to these shifts, often leading to a mismatch between regularization and update magnitude. In contrast, PEMA actively bounds this drift by linking β_t to the optimization trajectory. As shown in the right panel of Figure 13, PEMA adaptively adjusts the smoothing factor, ensuring regularization remains proportional to the actual distance traveled.

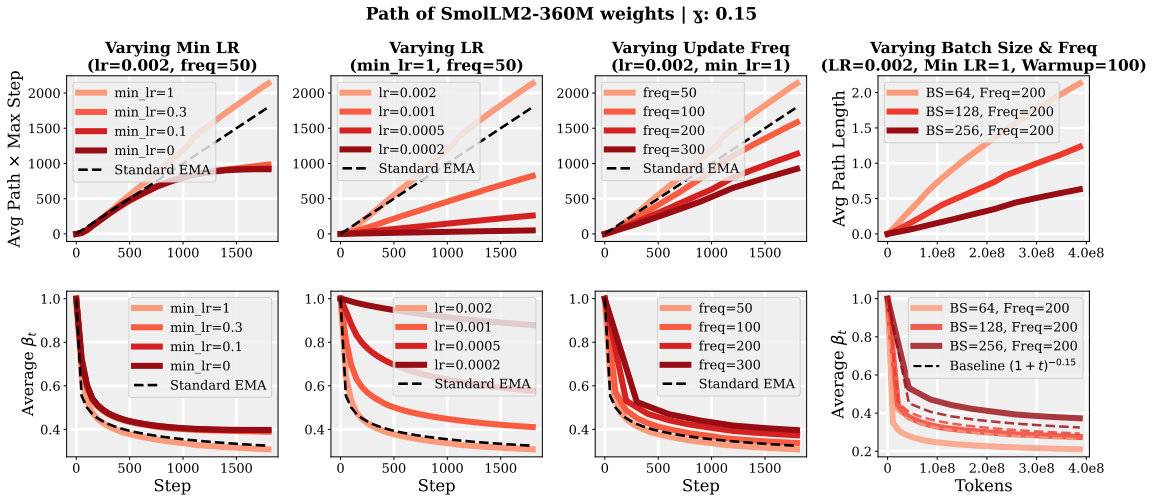

Figure 13: The average path traveled by model parameters across different hyperparameter configurations (left) and evolution of β_t with path length relative to token count (right). Note how the distance traveled by the weights increases as learning rate/minimum learning rate increases and as update frequency/batch size decreases, which leads to the algorithm making slower updates to the EMA weights. **PEMA automatically adapts to changing training conditions.** This adaptive mechanism ensures the model remains stable across disparate training configurations where standard fixed EMA often shows degraded performance.

C.6. Global vs. Per-Parameter Scaling

To evaluate the granularity of our scaling rule, we compared a global scaling approach (using a single β_t for the entire model) against a per-parameter approach. As shown in Table 2, performance is identical across configurations. This uniformity is explained by the observed standard deviation of path lengths across individual parameter tensors, which consistently remains below 2%. Consequently, parameters in modern Transformers move through the optimization manifold at highly correlated normalized speeds. For the per-parameter variant, we updated the i-th parameter using the following formulation:

$$\beta_{i,t} = \left(1 + T \cdot \ell(T)^2 \cdot \frac{\ell_i(t)}{\ell_i(T)}\right)^{-\gamma}$$

This equation was designed so that as training approaches the final step $t = T$, the per-parameter and global updates converge to the same value. However, it still allows for parameter-specific variations in $\beta_{i,t}$ during earlier stages of training to account for potential local trajectory differences.

Ultimately, we adopt the global formulation for Pathwise-EMA. Given the minimal performance difference, the global approach is significantly more efficient as it reduces memory overhead by removing the need to track unique path lengths for every individual weight tensor.

	0.002		0.001		0.0005		0.0002	
Min LR	global	ema	global	ema	global	ema	global	ema
0.0	0.989	0.989	1.004	1.004	1.049	1.049	1.124	1.124
0.1	0.989	0.989	1.004	1.004	1.049	1.049	1.124	1.124
0.3	0.985	0.985	1.001	1.001	1.047	1.047	1.122	1.122
1.0	1.018	1.018	0.994	0.994	1.017	1.017	1.085	1.085

Table 2: Comparison of final cross-entropy loss between global and per-parameter PEMA scaling. Both methods yield identical results due to low variance in path length across parameters.