ID BALANCING: STABLE TRAINING OF EXTREMELY SPARSE MOE VIA PID-BASED LOAD CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Scaling Large Language Models (LLMs) via Mixture-of-Experts (MoE) enables massive parameter growth with nearly constant per-token computation. However, further scaling the parameter count requires increasingly sparse routing, where expert load imbalance becomes more severe. This imbalance reduces parameter utilization and training efficiency, and can undermine training stability, becoming a bottleneck to reliable scaling. In this work, we unify two representative auxiliary-loss-free methods as incomplete Proportional–Integral–Derivative (PID) controllers: DeepSeek's loss-free method acts as a fixed-step integral controller, while Kimi K3's Quantile Balancing functions as a generalized proportional controller. Building on this control perspective, we propose ID Balancing, an Integral–Derivative controller. It scales its integral term with load error and activates its derivative term only when imbalance worsens, enabling stronger corrections for large or worsening errors and smaller updates near balance. Evaluated across Top-10, Top-5, and Top-3 routing over 768 experts, ID Balancing reduces worst-case backbone MaxVio and training-average backbone MinVio by over 50% and 12%, respectively, relative to the best baselines in the Top-3 setting. When scaling parameters from 18.9B to 69.9B (Top-10-of-768), its worst-case backbone MaxVio remains nearly unchanged and is approximately 89.6% lower than that of the auxiliary-loss baseline. ID Balancing also maintains competitive language-modeling and downstream performance. The advantages of ID Balancing grow as sparsity increases, making it a promising solution for scaling larger, sparser MoE models.

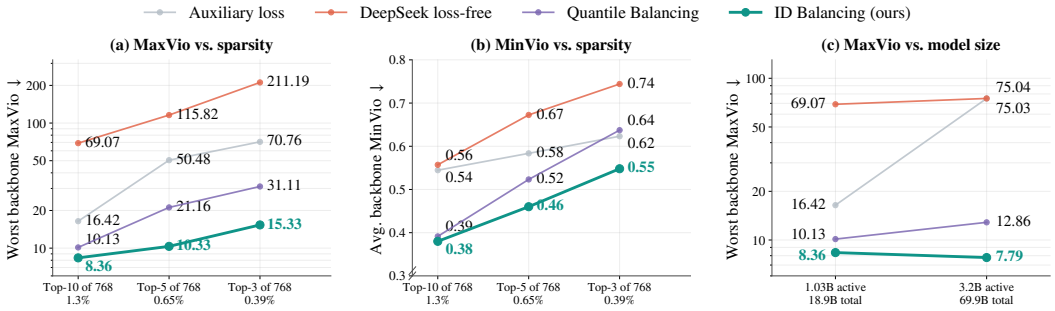


Figure 1: **ID Balancing maintains load control across routing sparsities and model sizes. (a, b)** Worst backbone MaxVio and training-average backbone MinVio, respectively, for Top-10, Top-5, and Top-3 routing over 768 experts in 18.9B models trained on 120B tokens. **(c)** Worst backbone MaxVio under Top-10-of-768 routing as total parameters increase from 18.9B to 69.9B (1.03B to 3.2B active), using the first 30k and 50k steps, respectively.

1 INTRODUCTION

Mixture-of-Experts (MoE) provides an efficient way to scale large language models (LLMs) through sparse expert activation (Jacobs et al., 1991; Roller et al., 2021). For each token, a router selects only K experts from a pool of E. With K fixed, adding experts increases total capacity without increasing the number of expert evaluations per token. This allows model capacity to grow under a fixed per-token compute budget. As the expert pool expands, the active fraction K/E decreases, making extremely sparse routing a natural direction for MoE scaling (Dai et al., 2024).

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

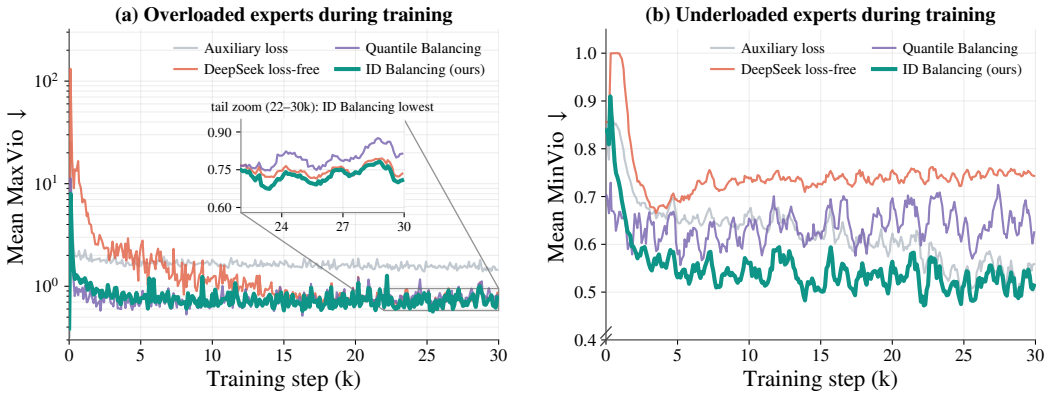


Figure 2: **ID Balancing limits transient overload and sustained underload.** Results use Top-3-of-768 routing in an 18.9B model trained on 120B tokens, averaged over 20 backbone layers. **(a)** Mean MaxVio on a logarithmic scale (↓). **(b)** Mean MinVio (↓).

As routing becomes sparser, maintaining balanced expert loads becomes more difficult. Small changes near the Top-K selection boundary alter token assignments, making expert loads sensitive to router scores (Lepikhin et al., 2021). Overloaded experts slow down MoE computation (He et al., 2022; 2021; Nie et al., 2022), while underloaded experts receive insufficient training and leave part of the model capacity unused. This imbalance limits both training efficiency and the benefits of adding more experts. Fig. 1 illustrates these challenges as routing becomes sparser or model capacity grows. Effective load control is therefore a key requirement for reliably training larger, sparser MoE models.

Existing auxiliary-loss-free methods (Wang et al., 2024b; Liu et al., 2024; Team et al., 2026) adjust expert biases using routing feedback. We interpret these updates through a unified Proportional–Integral–Derivative (PID) framework (Johnson & Moradi, 2005), where expert biases form the control state and deviations from uniform load provide the feedback signal. DeepSeek's loss-free method acts as a fixed-step integral controller, applying the same correction to small and large load errors. Kimi K3's Quantile Balancing acts as a generalized proportional controller, setting the bias from a target computed on the current batch without explicitly accumulating past load errors. Fig. 2 illustrates the transient overload and sustained underload observed under Top-3-of-768 routing. This control perspective links their update methods to their limitations and motivates a more adaptive response: *corrections should scale with the load error and respond when the imbalance worsens*.

Building on this control perspective, we propose ID Balancing, an **I**ntegral–**D**erivative controller for stable training of highly sparse MoE models. Its integral term applies larger corrections to larger load errors and smaller corrections near balance. Its derivative term uses changes in load error to add an extra correction only when the imbalance worsens. ID Balancing accumulates these corrections in the expert bias and omits the proportional term. Re-centering after each update removes the common bias component without changing routing decisions. The method requires only $\mathcal{O}(E)$ token-count feedback per layer and adds no balancing gradient to the language-modeling objective.

We evaluate ID Balancing with Top-10, Top-5, and Top-3 routing over 768 experts and model sizes up to 69.9B. We also test load control at a constant learning rate 2.3× the standard peak. In the Top-3 setting, ID Balancing reduces worst-case backbone MaxVio (Wang et al., 2024b) and training-average backbone MinVio by more than 50% and 12%, respectively, relative to the best baselines. Under fixed Top-10-of-768 routing, its worst-case backbone MaxVio remains nearly unchanged as total parameters increase from 18.9B to 69.9B. These load-control improvements are accompanied by competitive language-modeling and downstream performance.

Our contributions are: **(i)** We propose a unified PID view of auxiliary-loss-free MoE balancing. Under this view, DeepSeek's loss-free method acts as fixed-step integral control, while Quantile Balancing acts as generalized proportional control. **(ii)** We propose ID Balancing, combining a magnitude-aware integral term with a worsening-gated derivative term. The method adjusts expert biases using only $\mathcal{O}(E)$ token-count feedback per layer, without adding a balancing gradient to the language-modeling objective. **(iii)** We validate ID Balancing across routing sparsities, model scales, and training conditions. Further analyses show effective load control during continued pretraining, improved backbone load balance, and fewer inactive experts at inference.

2 Loss-Free Balancing as PID Control

2.1 The PID Controller

A Proportional–Integral–Derivative (PID) controller (Podlubny, 1994; Ang et al., 2005; Visioli, 2006; Shah & Agashe, 2016; Aström & Hägglund, 2005) uses feedback to drive a system toward a target. At step t, it computes the error e_t as the target minus the measured output and produces a control signal u_t. The proportional branch responds to the current error, the integral branch accumulates past errors, and the derivative branch responds to changes in error (Isaksson & Graebe, 2002). A standard discrete-time form is:

$$u_t = \underbrace{K_p\, e_t}_{\text{proportional}} + \underbrace{K_i \sum_{\tau \le t} e_\tau}_{\text{integral}} + \underbrace{K_d\,(e_t - e_{t-1})}_{\text{derivative}}, \tag{1}$$

where $K_p, K_i, K_d \ge 0$ are the gains of the three branches, and τ indexes the steps up to t. Eq. 1 is linear in the error. For MoE balancing, we use a generalized PID view that preserves the temporal roles of the three branches while allowing nonlinear feedback transformations. This view provides a common basis for comparing how expert-bias updates use feedback over time.

2.2 Load Balancing as a Control Problem

Consider an MoE layer with E routed experts and a Top-K router. For a token representation $\mathbf{x} \in \mathbb{R}^d$, the router computes logits $\mathbf{z} = W_r\mathbf{x}$, where d is the hidden size and $W_r \in \mathbb{R}^{E\times d}$ is the router weight matrix. An activation function σ maps each logit to an expert score $s_i = \sigma(z_i)$. Loss-free methods maintain a non-trainable bias b_i for each expert and select experts using:

$$\mathcal{T}(\mathbf{x}) = \mathrm{TopK}_{i\in[E]}(s_i + b_i)\,, \tag{2}$$

where $[E] = \{1, \ldots, E\}$ and $\mathcal{T}(\mathbf{x})$ contains the indices of the K largest biased scores. The router obtains combination weights by normalizing the selected experts' original scores s_i, without the biases. At training step t, let $n_i^{(t)}$ be the number of tokens assigned to expert i. The uniform target load and normalized load error are:

$$\bar{n}^{(t)} = \frac{1}{E}\sum_{i=1}^{E} n_i^{(t)}, \qquad e_i^{(t)} = \frac{\bar{n}^{(t)} - n_i^{(t)}}{\bar{n}^{(t)}}, \tag{3}$$

where $\bar{n}^{(t)} > 0$ for any nonempty batch. Negative errors indicate overload, while positive errors indicate underload. MaxVio and MinVio measure the largest deviations in these two directions, as defined in Eq. 12. These quantities define a feedback loop with uniform load as the target. The bias vector $\mathbf{b}^{(t)}$ serves as both the persistent controller state and the routing input, while the token-count vector $\mathbf{n}^{(t)}$ is the measured output. After each step, the controller uses the error vector $\mathbf{e}^{(t)}$ to update the bias, which changes routing decisions and expert loads at the next step.

2.3 Existing Methods through a PID Lens

We compare expert-bias updates by how they use routing feedback over time. Let $\mathbf{P}^{(t)}, \mathbf{I}^{(t)}, \mathbf{D}^{(t)} \in \mathbb{R}^E$ denote the proportional, integral, and derivative contributions. In a PID form, the next bias is:

$$\mathbf{b}^{(t+1)} = \mathbf{P}^{(t)} + \mathbf{I}^{(t)} + \mathbf{D}^{(t)}. \tag{4}$$

The proportional branch uses the current measurement or a target computed from the current batch. The integral branch accumulates past load errors, while the derivative branch uses the error change $\mathbf{e}^{(t)} - \mathbf{e}^{(t-1)}$. These temporal roles define the branches, allowing nonlinear feedback transformations rather than requiring the linear dependence in Eq. 1.

In MoE routing, the bias $\mathbf{b}^{(t)}$ is both a persistent controller state and the control input to the next Top-K decision. Eq. 4 uses the *position form*, which directly gives the next bias. The same update can also be written in an *incremental form* by adding $\Delta\mathbf{b}^{(t)} = \mathbf{b}^{(t+1)} - \mathbf{b}^{(t)}$ to the current bias. We use this generalized PID model as an interpretation of how a method uses feedback, not as a claim that every method follows the linear rule in Eq. 1. Under this view, DeepSeek loss-free maps to integral control, while Quantile Balancing can be interpreted as a generalized proportional controller.

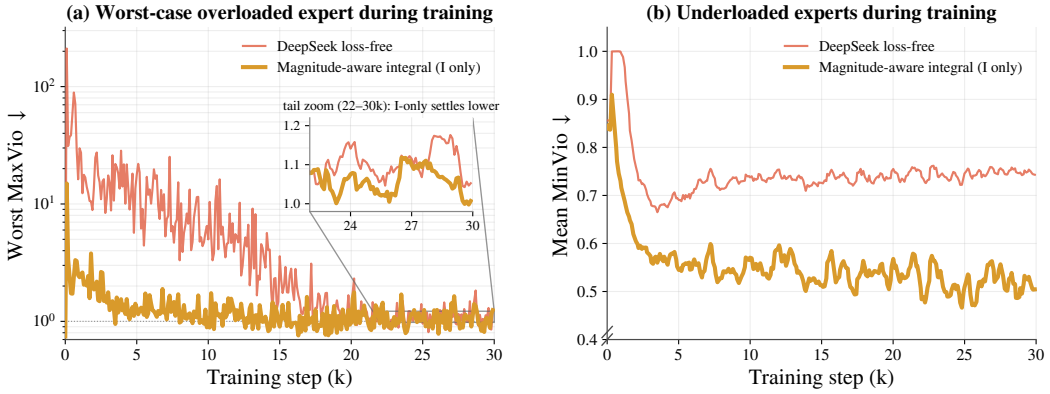


Figure 3: **Magnitude-aware integral control reduces early overload and sustained underload.** We compare the integral-only update with DeepSeek's loss-free method using Top-3-of-768 routing in an 18.9B model trained on 120B tokens. **(a)** Maximum MaxVio across 20 backbone layers on a logarithmic scale (↓). **(b)** Mean MinVio across the same layers (↓).

DeepSeek's loss-free method as fixed-step integral control. DeepSeek's loss-free method (Wang et al., 2024b; Liu et al., 2024) updates each expert bias according to the sign of its load error. Starting from $\mathbf{b}^{(0)} = \mathbf{0}$, the update is:

$$\mathbf{b}^{(t+1)} = \mathbf{b}^{(t)} + \eta \operatorname{sign}(\mathbf{e}^{(t)}) = \eta \sum_{\tau \le t} \operatorname{sign}(\mathbf{e}^{(\tau)}) = \mathbf{I}^{(t)}, \tag{5}$$

where $\eta > 0$ is the step size and $\operatorname{sign}(\cdot)$ acts elementwise. The update raises the bias of underloaded experts and lowers that of overloaded experts. The cumulative sum in Eq. 5 shows that the bias stores past sign errors. Under our generalized PID view, this corresponds to fixed-step integral control with $\mathbf{P}^{(t)} = \mathbf{D}^{(t)} = \mathbf{0}$. However, the sign operation discards error magnitude: every nonzero error receives a correction of size η, regardless of the severity of the imbalance.

Quantile Balancing as generalized proportional control. Quantile Balancing (Team et al., 2026) computes a target bias from the current batch's score distribution. Let $\mathbf{S}^{(t)} \in \mathbb{R}^{m \times E}$ contain the scores for m tokens, where $S_{i,j}^{(t)}$ is the score of expert j for token i. The method sets the next bias directly to this target:

$$\mathbf{b}^{(t+1)} = \mathcal{Q}(\mathbf{S}^{(t)}, \mathbf{b}^{(t)}) = \mathbf{P}^{(t)}. \tag{6}$$

The corresponding increment, $\mathcal{Q}(\mathbf{S}^{(t)}, \mathbf{b}^{(t)}) - \mathbf{b}^{(t)}$, corrects the deviation from the current target.

Following Kimi K3, the target assigns approximately $q = mK/E$ tokens to each expert. For token i, let $\alpha_i^{(t)}$ be the $(K{+}1)$-th largest biased score $S_{i,r}^{(t)} + b_r^{(t)}$ over $r \in [E]$. The target bias for expert j is:

$$\mathcal{Q}_j(\mathbf{S}^{(t)}, \mathbf{b}^{(t)}) = -\operatorname{quantile}_{1-K/E}\left(\{\, S_{i,j}^{(t)} - \alpha_i^{(t)} \,\}_{i=1}^{m}\right), \tag{7}$$

where $S_{i,j}^{(t)} - \alpha_i^{(t)}$ is the score margin relative to the current cutoff. With these cutoffs held fixed, the target places approximately K/E of the biased margins above zero, matching the desired load q. Ties and integer rounding affect the exact count. The target changes with the batch score distribution and depends on the current bias through $\alpha_i^{(t)}$. The update therefore retains state dependence, but does not explicitly accumulate past load errors. Under our generalized PID view, this direct response to a batch-dependent target forms a nonlinear proportional branch with $\mathbf{I}^{(t)} = \mathbf{D}^{(t)} = \mathbf{0}$.

Auxiliary loss as gradient-based balancing. Auxiliary-loss methods (Shazeer et al., 2017; Lepikhin et al., 2021; Xue et al., 2024) encourage balanced expert loads by adding a differentiable term to the training objective. For a batch of T tokens, a common form is:

$$\mathcal{L}_{\text{aux}} = \alpha \sum_{i=1}^{E} f_i \, P_i, \qquad f_i = \frac{1}{T} \sum_{\mathbf{x}} \mathbb{1}[i \in \mathcal{T}(\mathbf{x})], \qquad P_i = \frac{1}{T} \sum_{\mathbf{x}} p_i(\mathbf{x}), \tag{8}$$

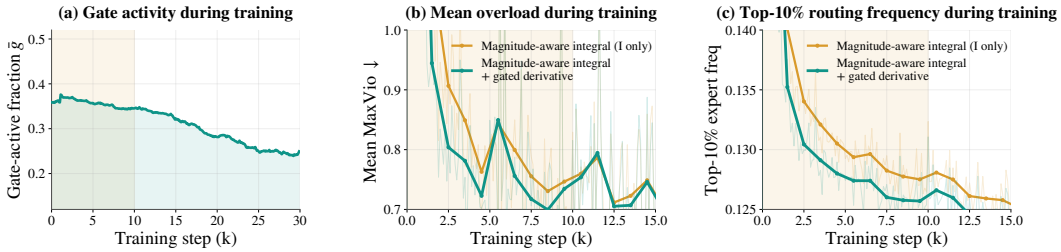


Figure 4: **The worsening-gated derivative term reduces early overload and expert concentration.** Results use Top-3-of-768 routing in an 18.9B model trained on 120B tokens. Metrics average over 20 backbone layers. **(a)** $\bar{g}^{(t)}$ is the fraction of active gates. **(b)** Mean MaxVio (↓). **(c)** Assignment share of the busiest 10% of experts ($\rightarrow 0.10$), with a balanced target of 0.10.

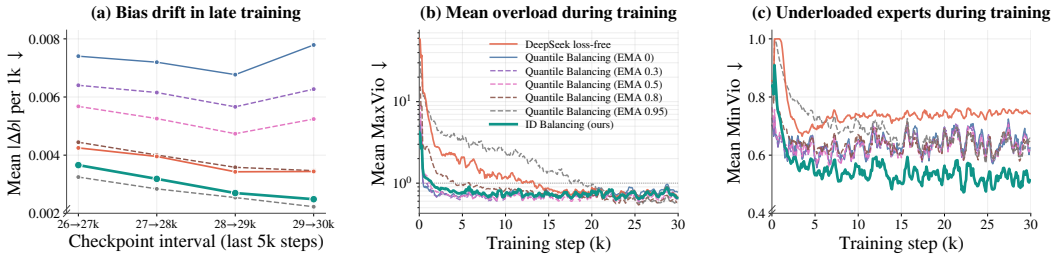


Figure 5: **EMA smoothing reduces bias drift but slows load correction in Quantile Balancing.** Results use Top-3-of-768 routing in an 18.9B model trained on 120B tokens. Here, $\rho = 0$ is the original update. **(a)** Mean absolute bias change between checkpoints 1k steps apart during the final 5k steps (↓). **(b)** Mean MaxVio on a logarithmic scale (↓). **(c)** Mean MinVio (↓).

where the sums cover all tokens in the batch and $\mathbb{1}[\cdot]$ is the indicator function. Here, f_i is the fraction of tokens assigned to expert i, and P_i is its average router probability, with $p_i(\mathbf{x}) = \mathrm{softmax}_i(W_r\mathbf{x})$. The coefficient α controls the loss strength and absorbs constant scaling factors. The balancing loss influences token assignments through gradients, without maintaining a separate control bias. This places auxiliary-loss balancing outside the expert-bias control family in Eq. 4.

3 ID Balancing

ID Balancing changes only the expert-bias update, retaining the routing rule in Eq. 2 and the normalized error in Eq. 3. The controller combines a magnitude-aware integral term with a derivative term that activates only when the imbalance worsens. It omits the proportional term and re-centers the bias after each update to remove its common component. Algorithm 1 in Appendix A.5 gives the complete update.

Magnitude-aware integral term. DeepSeek's loss-free method uses only the sign of the load error, giving every nonzero error the same correction size. To adapt the response to the severity of the imbalance, we scale the update by the normalized error itself:

$$b_i^{(t+1)} = b_i^{(t)} + K_i\, e_i^{(t)}, \tag{9}$$

where $K_i \geq 0$ is the integral gain shared by all experts. The correction grows with $|e_i^{(t)}|$ and shrinks as the expert approaches its target load. Fig. 3 compares the two integral updates. Scaling by error magnitude reduces the worst-layer MaxVio peak and lowers sustained underload, showing stronger early correction and better expert utilization.

Worsening-gated derivative term. The integral term responds to error magnitude but does not distinguish growing from shrinking errors of the same size. To capture this difference, we compute the change in error $\Delta e_i^{(t)} = e_i^{(t)} - e_i^{(t-1)}$, where $e_i^{(-1)} = 0$. We retain only changes that move an existing imbalance farther from zero:

$$g_i^{(t)} = \mathbb{1}\left[\, e_i^{(t-1)}\, \Delta e_i^{(t)} > 0 \,\right] \in \{0, 1\}, \tag{10}$$

Table 1: **ID Balancing improves backbone load control.** Backbone Last1k and Avg. metrics average over 20 layers, while Worst MaxVio takes the maximum across layers and steps.

Method	Act./Total	Backbone						MTP Module				
		LM	MaxVio ↓			MinVio ↓		MaxVio ↓			MinVio ↓	
		loss	Last1k	Avg.	Worst	Last1k	Avg.	Last1k	Avg.	Worst	Last1k	Avg.
Top-3 of 768 experts, 120B tokens												
Auxiliary loss	0.87B / 18.9B	1.7527	1.4809	1.7239	70.76	0.5462	0.6232	3.8952	4.4639	43.38	0.6580	0.7502
DeepSeek loss-free		**1.7426**	0.7314	1.9356	211.19	0.7487	0.7441	**0.7477**	**0.8930**	**17.32**	0.6868	0.6533
Quantile Balancing		1.7432	0.8093	0.7985	31.11	0.6504	0.6373	1.8777	1.2071	30.89	0.7106	0.7083
ID Balancing		**1.7426**	**0.7026**	**0.7931**	**15.33**	**0.5266**	**0.5480**	1.0165	0.9698	33.42	**0.5832**	**0.5782**
Top-5 of 768 experts, 120B tokens												
Auxiliary loss	0.91B / 18.9B	1.7378	1.0296	1.2925	50.48	0.4537	0.5836	2.8407	3.1892	42.60	0.6006	0.6851
DeepSeek loss-free		**1.7280**	0.5821	1.4814	115.82	0.6741	0.6725	**0.6515**	**0.7284**	15.75	0.5517	0.5376
Quantile Balancing		1.7303	0.6376	**0.6394**	21.16	0.5335	0.5232	1.2480	0.8683	**7.73**	0.5516	0.5698
ID Balancing		1.7299	**0.5404**	0.6498	**10.33**	**0.4238**	**0.4601**	0.7808	0.7710	28.76	**0.4754**	**0.4750**
Top-10 of 768 experts, 120B tokens												
Auxiliary loss	1.03B / 18.9B	1.7180	0.6862	1.0342	16.42	0.3631	0.5446	2.0145	2.2781	22.74	0.4511	0.5589
DeepSeek loss-free		1.7121	0.4418	0.9715	69.07	0.5534	0.5571	**0.5828**	0.6066	14.79	0.4752	0.4237
Quantile Balancing		1.7133	0.4776	**0.4832**	10.13	0.4060	0.3914	0.8642	0.6878	**5.16**	**0.4443**	0.4263
ID Balancing		**1.7118**	**0.3863**	0.5110	**8.36**	**0.3282**	**0.3800**	0.6070	**0.5942**	21.98	0.4938	**0.3842**

where $\mathbb{1}[\cdot]$ is the indicator function. The gate opens when the previous error and its change have the same sign, so the imbalance grows without crossing zero. Fig. 4 shows that adding the gated correction lowers early load violations, while the trajectories approach those of the integral-only update as training progresses.

Why ID Balancing omits the proportional term. ID Balancing sets $K_p = 0$ and controls expert loads through accumulated bias corrections. Quantile Balancing follows a target computed from the current batch. Changes in the score distribution lead to larger bias changes. Smaller changes in relative expert biases limit perturbations to the Top-K selection boundary, helping stabilize routing. Smoother bias trajectories are also desirable for weight merging, such as averaging nearby checkpoints, where the merged weights must remain compatible with the routing state. Lower bias drift may reduce this mismatch risk by keeping the routing states of the source checkpoints closer. To examine the trade-off between bias smoothing and load control, we add exponential moving average (EMA) smoothing (Winters, 1960; Hunter, 1986) to Quantile Balancing only as a diagnostic: $\mathbf{b}^{(t+1)} = \rho\,\mathbf{b}^{(t)} + (1-\rho)\,\mathcal{Q}(\mathbf{S}^{(t)}, \mathbf{b}^{(t)})$. Here, $0 \le \rho < 1$ controls the smoothing strength, and $\rho = 0$ recovers the original update without EMA. Fig. 5 shows that stronger smoothing reduces bias drift but weakens early load control. ID Balancing achieves low bias drift and effective load balance through its integral and gated derivative updates, without additional EMA smoothing.

Zero-mean centering step. The integral update has zero mean because $\sum_i e_i^{(t)} = 0$. Gating does not preserve this property: $\sum_i g_i^{(t)} \Delta e_i^{(t)}$ is generally nonzero, allowing the mean bias to drift as derivative corrections accumulate. Adding the same constant to every expert bias leaves the score ranking and the selected set $\mathcal{T}(\mathbf{x})$ unchanged. We therefore remove this common component after each update. It removes common bias drift but does not bound the differences between expert biases.

The complete ID Balancing update. Combining the integral and gated derivative corrections with re-centering gives:

$$\widetilde{b}_i^{(t+1)} = b_i^{(t)} + \underbrace{K_i\, e_i^{(t)}}_{\text{magnitude-aware integral}} + \underbrace{K_d\, g_i^{(t)}\, \Delta e_i^{(t)}}_{\text{worsening-gated derivative}}, \quad b_i^{(t+1)} = \widetilde{b}_i^{(t+1)} - \frac{1}{E}\sum_{j=1}^{E} \widetilde{b}_j^{(t+1)}. \tag{11}$$

where $\widetilde{b}_i^{(t+1)}$ is the bias before re-centering and $K_d \ge 0$ is the derivative gain. The centering step ensures $\sum_i b_i^{(t+1)} = 0$. The resulting bias controls routing at the next step.

4 Experiments

4.1 Model Quality

Small-model validation. Our main comparison uses the Qwen-3.8-Next architecture (Qiu et al., 2026) with 20 Transformer layers, one multi-token prediction (MTP) module (Gloeckle et al., 2024),

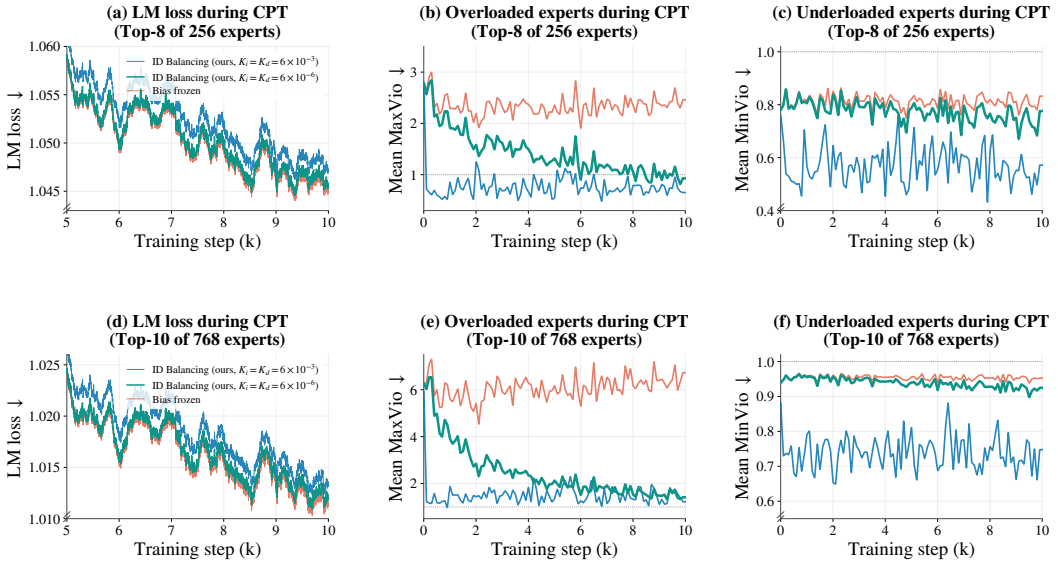


Figure 6: **Reduced ID gains retain load control and competitive LM loss during continued pretraining.** We compare $K_i = K_d \in \{6 \times 10^{-3}, 6 \times 10^{-6}\}$ and frozen biases. **(a, d)** LM loss (↓). **(b, e)** Mean MaxVio (↓). **(c, f)** Mean MinVio (↓).

Table 2: **ID Balancing maintains competitive downstream performance.** Results use Top-8-of-256 and Top-10-of-768 routing. Avg. is the mean across nine benchmarks.

Method	Knowledge			STEM		Reasoning	Multilingual	Code		Avg.
	MMLU	MMLU-Pro	SuperGPQA	MATH	GSM8K	BBH	MMMLU	EvalPlus	MultiPL-E	
Top-8 of 256 experts, 3.0B / 24.8B, 560B tokens										
Auxiliary loss	68.00	47.10	27.36	47.12	74.94	69.57	59.82	51.51	**46.22**	54.63
DeepSeek loss-free	70.31	47.47	27.49	46.44	**77.56**	70.31	**60.71**	53.63	46.04	55.55
Quantile Balancing	**70.40**	47.93	**27.91**	**47.48**	73.62	70.53	60.18	51.81	44.28	54.90
ID Balancing	69.48	**48.26**	27.17	47.40	75.36	**73.88**	60.62	**53.99**	44.33	**55.61**
Top-10 of 768 experts, 3.2B / 69.9B, 560B tokens										
Auxiliary loss	71.67	**50.91**	30.15	50.74	79.08	71.09	63.62	56.55	49.28	58.12
DeepSeek loss-free	72.23	49.55	30.19	51.00	77.94	72.29	**64.07**	**57.61**	**51.90**	58.53
Quantile Balancing	**72.74**	50.40	**30.72**	50.40	77.37	72.28	63.97	57.55	50.92	58.48
ID Balancing	71.84	50.61	30.08	**51.04**	**79.34**	**73.52**	63.72	56.51	50.82	**58.61**

and 768 routed experts per MoE layer. Top-3, Top-5, and Top-10 variants have 18.9B total and 0.87–1.03B active parameters. They are trained on 120B tokens over 30k steps with a global batch size of 1024. The learning rate peaks at 2.54×10^{-3} and follows cosine decay to 3×10^{-5}. Full architecture specifications appear in Appendix A.1.

We compare Auxiliary loss, DeepSeek's loss-free method, Quantile Balancing, and ID Balancing using the same architecture, data, and optimizer settings. ID Balancing uses $K_i = K_d = 6 \times 10^{-3}$, with gain ablations in Section 4.4. Quantile Balancing uses no EMA except in Fig. 5. We measure overload and underload relative to the uniform target $\bar{n}^{(t)}$ in Eq. 3:

$$\mathrm{MaxVio}^{(t)} = \frac{\max_i n_i^{(t)} - \bar{n}^{(t)}}{\bar{n}^{(t)}}, \qquad \mathrm{MinVio}^{(t)} = \frac{\bar{n}^{(t)} - \min_i n_i^{(t)}}{\bar{n}^{(t)}}, \tag{12}$$

Here, $\max_i n_i^{(t)}$ and $\min_i n_i^{(t)}$ are the largest and smallest numbers of tokens assigned to any of the E routed experts in the layer at step t, respectively. MaxVio measures the largest relative overload. MinVio measures the largest relative deficit and lies in $[0, 1]$, reaching 1 when an expert receives no tokens. Last1k and Avg. denote means over the final 1k steps and the full training window. Backbone values average over 20 layers, except Worst MaxVio, which takes the maximum over layers and steps. LM loss averages over the final 1k steps. Act./Total lists active and total parameter counts. Tab. 1 shows similar LM loss among the three bias-based methods, with a maximum gap of 0.0023, while Auxiliary loss has the highest loss in every setting. ID Balancing achieves the lowest backbone Worst and Last1k MaxVio and Last1k and Avg. MinVio across all three routing settings.

Table 3: **ID Balancing and Quantile Balancing maintain load control at a higher learning rate.** Results use a 1.03B-active/18.9B-total model with Top-10-of-768 routing and 120B training tokens at a constant learning rate of 5.86×10^{-3} (2.3× the standard peak).

Method	Backbone						MTP Module				
	LM	MaxVio ↓			MinVio ↓		MaxVio ↓			MinVio ↓	
	loss	Last1k	Avg.	Worst	Last1k	Avg.	Last1k	Avg.	Worst	Last1k	Avg.
Auxiliary loss	2.0099	3.1383	2.8379	20.89	0.8423	0.8458	3.5640	3.0533	**4.83**	0.6569	0.7130
DeepSeek loss-free	2.0031	1.3755	1.7811	72.54	0.8730	0.8269	0.5033	0.7124	17.98	0.6105	0.5514
Quantile Balancing	2.0009	**0.5021**	**0.4979**	**2.96**	**0.4083**	**0.3938**	0.6243	0.5836	10.77	0.4346	0.4308
ID Balancing	**2.0007**	0.6437	0.7274	11.97	0.4828	0.4950	**0.4642**	**0.5068**	7.20	**0.3940**	**0.3927**

Table 4: $K_i = 6 \times 10^{-3}$ **yields the lowest worst-case backbone MaxVio among the tested gains.** Results use a 0.87B-active/18.9B-total model with Top-3-of-768 routing over 30k steps, fixing $K_d = 0$ and varying $K_i \in \{3, 6, 9\} \times 10^{-3}$. Lower values are better for all metrics.

Setting	Backbone						MTP module				
	LM	MaxVio ↓			MinVio ↓		MaxVio ↓			MinVio ↓	
	loss	Last1k	Avg.	Worst	Last1k	Avg.	Last1k	Avg.	Worst	Last1k	Avg.
Integral gain K_i, integral term only, K_d=0											
$K_i = 3 \times 10^{-3}$	**1.7422**	**0.6759**	0.9459	23.26	**0.5190**	0.5777	**0.6375**	**0.7991**	**9.14**	**0.5603**	0.5775
$K_i = 6 \times 10^{-3}$	1.7431	0.6924	0.8003	**14.96**	0.5233	0.5527	0.8557	0.8862	15.18	0.5848	**0.5714**
$K_i = 9 \times 10^{-3}$	1.7429	0.6831	**0.7872**	20.13	0.5263	**0.5436**	0.9222	1.0388	35.56	0.6279	0.5800

Maintaining active load control during continued pretraining. Freezing expert biases removes active load correction during continued pretraining (Liu et al., 2024), allowing loads to drift as the data distribution changes (Gururangan et al., 2020; Ke et al., 2023; Gupta et al., 2023). We compare full ID gains ($K_i = K_d = 6 \times 10^{-3}$), gains reduced by 1000× (6×10^{-6}), and a frozen bias under Top-8-of-256 and Top-10-of-768 routing. Fig. 6 shows that full gains yield the highest LM loss, while freezing leaves substantial imbalance. Reduced gains keep LM loss within 0.001 of the frozen baseline during the second half of continued pretraining, with mean MaxVio near 1 and 1.4 in the two settings, respectively. We use these reduced gains ($K_i = K_d = 6 \times 10^{-6}$) for continued pretraining. The DeepSeek and Quantile Balancing baselines keep their biases frozen during this stage. Appendix B.1 provides the settings and detailed comparison.

Downstream quality. We evaluate models trained on 560B tokens with Top-8-of-256 and Top-10-of-768 routing across nine benchmarks covering knowledge (Hendrycks et al., 2021a; Wang et al., 2024c; Du et al., 2026), mathematics (Hendrycks et al., 2021b; Cobbe et al., 2021), reasoning (Suzgun et al., 2023), multilingual understanding (OpenAI, 2024), and code generation (Cassano et al., 2023). Tab. 2 shows an average score of 55.61 for ID Balancing under Top-8-of-256 routing. Under Top-10-of-768 routing, it improves the average from 58.12 to 58.61 relative to Auxiliary loss and scores higher than this baseline on six of the nine benchmarks.

4.2 Training Stability

Load control at a higher learning rate. At a constant learning rate of 5.86×10^{-3} (2.3× the standard peak), Tab. 3 shows lower backbone violations for ID Balancing than for Auxiliary loss and DeepSeek's loss-free method. ID Balancing has the lowest average MTP overload and competitive LM loss. Appendix B.2 provides the full trajectories and auxiliary-loss coefficient analysis. ID Balancing shows smooth gradient-norm trajectories. Appendix B.4 examines these trajectories and the corresponding MoE-output magnitudes.

4.3 Model Characteristics

Training-time characteristics. Under Top-3-of-768 routing, ID Balancing achieves lower or comparable MaxVio and lower MinVio than Quantile Balancing across all 20 backbone layers. Appendix B.3 reports the layer-wise profiles and depth trends.

Table 5: **Larger derivative gains reduce backbone overload over 1–5k steps but increase MTP overload.** Results use a 0.87B-active/18.9B-total model with Top-3-of-768 routing over 30k steps. We fix $K_i = 6 \times 10^{-3}$ and vary $K_d \in \{0, 3, 6, 12\} \times 10^{-3}$, where $K_d = 0$ is the integral-only baseline. Avg. denotes full-training means.

Setting	Backbone							MTP module			
	LM loss	0–1k steps		1–5k steps		Avg.		0–5k steps		Avg.	
		MaxVio ↓	MinVio ↓	MaxVio ↓	MinVio ↓	MaxVio ↓	MinVio ↓	MaxVio ↓	MinVio ↓	MaxVio ↓	MinVio ↓
Derivative gain K_d, fixed $K_i = 6 \times 10^{-3}$											
$K_d = 0$ (I only)	1.7431	2.0020	0.7955	0.9059	0.5989	0.8003	0.5527	**1.4965**	0.6222	**0.8862**	0.5714
$K_d = 3 \times 10^{-3}$	1.7428	1.8900	0.7853	0.8869	0.5992	0.7910	0.5485	1.7454	0.6201	0.9459	0.5799
$K_d = 6 \times 10^{-3}$	**1.7426**	1.9330	0.7791	0.8612	0.5899	0.7931	**0.5480**	1.7724	0.6336	0.9698	0.5782
$K_d = 12 \times 10^{-3}$	1.7427	**1.8100**	**0.7644**	**0.8131**	**0.5833**	**0.7812**	0.5501	2.0382	**0.6048**	1.0922	**0.5660**

Expert utilization at inference. ID Balancing models leave fewer experts unused on evaluation data than Auxiliary loss. Under Top-10-of-768 routing, the average inactive-expert ratio falls from 8.4% to 6.4%. Appendix B.5 provides layer-wise utilization results and routing-score entropy analysis.

4.4 Ablations

Integral gain K_i in isolation. Tab. 4 shows that increasing K_i from 3×10^{-3} to 6×10^{-3} reduces worst-case backbone MaxVio from 23.26 to 14.96. The smallest gain retains the lowest LM loss, while 9×10^{-3} increases MTP load violations.

Derivative gain K_d and early load control. With $K_i = 6 \times 10^{-3}$, the default $K_d = 6 \times 10^{-3}$ improves early backbone balance relative to I only (Tab. 5). The full-training backbone Avg. MaxVio improvement is modest, from 0.8003 to 0.7931, while MTP Avg. MaxVio rises from 0.8862 to 0.9698. Appendix B.6.2 details this trade-off through the trajectories and cumulative excess overload.

5 Related Work

Sparse mixture-of-experts. Building on Transformer-based language modeling (Vaswani et al., 2017; Radford et al., 2019; Brown et al., 2020) and instruction-tuned and conversational systems (Ouyang et al., 2022; OpenAI, 2022; Achiam et al., 2023), sparse MoE increases capacity through conditional expert activation (Du et al., 2022; Rajbhandari et al., 2022; Muennighoff et al., 2025; Puigcerver et al., 2024; He, 2024; Yang et al., 2025a). Switch Transformer (Fedus et al., 2022) and GShard (Lepikhin et al., 2021) use Top-1 and Top-2 routing, while BASE Layers (Lewis et al., 2021), Hash Layers (Roller et al., 2021), and Expert Choice (Zhou et al., 2022) explore alternative assignments. MoH (Jin et al., 2025b) applies sparse activation to attention heads, and MoE++ (Jin et al., 2025a) combines feed-forward and zero-computation experts.

Balancing through an auxiliary loss. Auxiliary losses encourage balanced token assignments through router probabilities (Fedus et al., 2022). Their gradients update the same router weights as the language-modeling objective, introducing a multi-objective trade-off (Yu et al., 2020; Sener & Koltun, 2018; Liu et al., 2021). LocMoE (Li et al., 2024) and MegaBlocks (Gale et al., 2023) further address efficient MoE execution.

Auxiliary-loss-free balancing. Auxiliary-loss-free methods adjust expert biases for Top-K selection without including them in combination weights (Wang et al., 2024b; Liu et al., 2024). DeepSeek's loss-free method accumulates fixed-magnitude sign corrections, while Kimi K3's Quantile Balancing (Team et al., 2026) follows a target computed from the current batch's score margins. We interpret these updates as fixed-step integral and generalized proportional control, respectively.

6 Conclusion

We propose a generalized PID view of auxiliary-loss-free balancing and develop ID Balancing through magnitude-aware integral control and a worsening-gated derivative correction. The method uses $\mathcal{O}(E)$ token-count feedback per layer without adding a balancing gradient. Experiments show improved backbone load control with competitive language-modeling and downstream performance. Reduced gains retain active correction during continued pretraining. Further analyses show consistent backbone load control across layers and fewer inactive experts on evaluation data.

AI USE STATEMENT

We used generative AI to improve the clarity and wording of the manuscript. We manually reviewed the AI-edited text to ensure that it accurately and rigorously conveys our methods, results, and conclusions.

ETHICS STATEMENT

This work studies optimization and routing methods for mixture-of-experts language models. It does not introduce new datasets involving human subjects or new user-facing deployment capabilities. Its broader impacts include the computational and energy costs of large-scale language-model training.

REPRODUCIBILITY STATEMENT

The method is defined in Sections 2 and 3, with the complete bias-update procedure in Appendix A.5. Experimental settings, metrics, and quantitative comparisons are provided in Section 4. Architecture specifications, training settings, metric aggregation, and benchmark references appear in Appendix A. Detailed trajectory and utilization analyses appear in Appendix B.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Kiam Heong Ang, Gregory Chong, and Yun Li. Pid control system analysis, design, and technology. *IEEE transactions on control systems technology*, 13(4):559–576, 2005.

Karl J Aström and Tore Hägglund. *Advanced PID control*. John Wiley & Sons, 2005.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, pp. 1877–1901, 2020.

Federico Cassano, John Gouwar, Daniel Nguyen, Sydney Nguyen, Luna Phipps-Costin, Donald Pinckney, Ming-Ho Yee, Yangtian Zi, Carolyn Jane Anderson, Molly Q Feldman, et al. Multipl-e: A scalable and polyglot approach to benchmarking neural code generation. *IEEE Transactions on Software Engineering*, 49(7):3675–3691, 2023.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *ACL*, pp. 1280–1297, 2024.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. Glam: Efficient scaling of language models with mixture-of-experts. In *ICML*, pp. 5547–5569, 2022.

Xeron Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, et al. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. In *NeurIPS*, 2026.

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

Trevor Gale, Deepak Narayanan, Cliff Young, and Matei Zaharia. Megablocks: Efficient sparse training with mixture-of-experts. In *MLSys*, pp. 288–304, 2023.

Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction. *arXiv preprint arXiv:2404.19737*, 2024.

Kshitij Gupta, Benjamin Thérien, Adam Ibrahim, Mats L Richter, Quentin Anthony, Eugene Belilovsky, Irina Rish, and Timothée Lesort. Continual pre-training of large language models: How to (re) warm your model? *arXiv preprint arXiv:2308.04014*, 2023.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *ACL*, pp. 8342–8360, 2020.

Jiaao He, Jiezhong Qiu, Aohan Zeng, Zhilin Yang, Jidong Zhai, and Jie Tang. Fastmoe: A fast mixture-of-expert training system. *arXiv preprint arXiv:2103.13262*, 2021.

Jiaao He, Jidong Zhai, Tiago Antunes, Haojie Wang, Fuwen Luo, Shangfeng Shi, and Qin Li. Fastermoe: modeling and optimizing training of large-scale dynamic pre-trained models. In *PPoPP*, pp. 120–134, 2022.

Xu Owen He. Mixture of a million experts. *arXiv preprint arXiv:2407.04153*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *ICLR*, 2021a.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021b.

J Stuart Hunter. The exponentially weighted moving average. *Journal of quality technology*, 18(4): 203–210, 1986.

AJ Isaksson and SF Graebe. Derivative filter is an integral part of pid design. *IEE Proceedings-Control Theory and Applications*, 149(1):41–45, 2002.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Peng Jin, Bo Zhu, Yuan Li, and Shuicheng Yan. Moe++: Accelerating mixture-of-experts methods with zero-computation experts. In *ICLR*, pp. 50832–50856, 2025a.

Peng Jin, Bo Zhu, Li Yuan, and Shuicheng Yan. Moh: Multi-head attention as mixture-of-head attention. In *ICML*, 2025b.

Michael A Johnson and Mohammad H Moradi. *PID control*. Springer, 2005.

Zixuan Ke, Yijia Shao, Haowei Lin, Tatsuya Konishi, Gyuhak Kim, and Bing Liu. Continual pre-training of language models. *arXiv preprint arXiv:2302.03241*, 2023.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In *ICLR*, 2021.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. Base layers: Simplifying training of large, sparse models. In *ICML*, pp. 6265–6274, 2021.

Jing Li, Zhijie Sun, Xuan He, Li Zeng, Yi Lin, Entong Li, Binfan Zheng, Rongqian Zhao, and Xin Chen. Locmoe: A low-overhead moe for large language model training. *arXiv preprint arXiv:2401.13920*, 2024.

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.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. In *NeurIPS*, pp. 18878–18890, 2021.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, et al. Olmoe: Open mixture-of-experts language models. In *ICLR*, pp. 62061–62121, 2025.

Xiaonan Nie, Pinxue Zhao, Xupeng Miao, Tong Zhao, and Bin Cui. Hetumoe: An efficient trillion-scale mixture-of-expert distributed training system. *arXiv preprint arXiv:2203.14685*, 2022.

OpenAI. Introducing chatgpt. *CoRR*, 2022. URL https://openai.com/blog/chatgpt.

OpenAI. Multilingual massive multitask language understanding (mmmlu). *Dataset available at Hugging Face*, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, pp. 27730–27744, 2022.

Igor Podlubny. Fractional-order systems and fractional-order controllers. *Institute of Experimental Physics, Slovak Academy of Sciences, Kosice*, 12(3):1–18, 1994.

Joan Puigcerver, Carlos Riquelme Ruiz, Basil Mustafa, and Neil Houlsby. From sparse to soft mixtures of experts. In *ICLR*, pp. 28435–28445, 2024.

Zihan Qiu, Zekun Wang, Xiao Li, Yanpeng Li, Yang Xu, Yixuan Wang, Huaqing Zhang, Rui Men, Bochao Mao, Chengruidong Zhang, et al. On the design of qwen3. 8-next architecture: Evaluation, efficiency, and training stability. *arXiv preprint arXiv:2608.30320*, 2026.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. In *ICML*, pp. 18332–18346, 2022.

Stephen Roller, Sainbayar Sukhbaatar, Jason Weston, et al. Hash layers for large sparse models. In *NeurIPS*, pp. 17555–17566, 2021.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *NeurIPS*, 2018.

Pritesh Shah and Sudhir Agashe. Review of fractional pid controller. *Mechatronics*, 38:29–41, 2016.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the ACL*, pp. 13003–13051, 2023.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Antonio Visioli. *Practical PID control*. Springer, 2006.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(10):6761–6774, 2024a.

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *arXiv preprint arXiv:2408.15664*, 2024b.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *NeurIPS*, pp. 95266–95290, 2024c.

Peter R Winters. Forecasting sales by exponentially weighted moving averages. *Management science*, 6(3):324–342, 1960.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *ICML*, pp. 10524–10533, 2020.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. Openmoe: An early effort on open mixture-of-experts language models. *arXiv preprint arXiv:2402.01739*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *ICML*, pp. 56501–56523, 2024.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *ICLR*, pp. 29687–29707, 2025b.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *NeurIPS*, pp. 5824–5836, 2020.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. Mixture-of-experts with expert choice routing. In *NeurIPS*, pp. 7103–7114, 2022.

Table A: **Architectural specifications for the evaluated MoE models.** Parameter counts include the input embedding and untied output head, and exclude the MTP module. Active parameters include all dense and shared components together with the routed experts selected by Top-K.

Specification	Small MoE	Medium MoE	Large MoE
Transformer layers	20	28	28
Hidden size	1024	2048	2048
Vocabulary size	248320	248320	248320
Softmax-attention heads	16	16	16
Query groups	2	2	2
Softmax-attention head dimension	256	256	256
Full-attention interval	4	4	4
Linear-attention key/value dimension	128 / 128	128 / 128	128 / 128
Linear-attention key/value heads	8 / 32	16 / 32	16 / 32
Per-expert FFN size	384	512	512
Routed experts	768	256	768
Shared experts	1	1	1
Top-K	3 / 5 / 10	8	10
Tied input/output embeddings	No	No	No
Total parameters	18.9B	24.8B	69.9B
Active parameters	0.87B / 0.91B / 1.03B	3.0B	3.2B

A Experimental Setup

This appendix specifies the model architectures, training settings, metric aggregation, downstream benchmarks, and bias-update procedure used in Section 4.

A.1 Model architectures

We evaluate three hybrid MoE configurations that cover different expert-pool and model scales. Their specifications are summarized in Tab. A. All three models combine full attention and gated linear attention (Yang et al., 2024; 2025b; Dao & Gu, 2024; Sun et al., 2023), with one full-attention layer every four Transformer layers. Each MoE layer contains one shared expert in addition to the routed experts. The input embedding and output head are untied.

A.2 Training configurations

Standard pretraining. The main comparison uses the Qwen-3.8-Next architecture (Qiu et al., 2026) with 20 Transformer layers, one multi-token prediction (MTP) module (Gloeckle et al., 2024), and 768 routed experts per MoE layer. Top-3, Top-5, and Top-10 routing gives 0.87B, 0.91B, and 1.03B active parameters, respectively, with 18.9B total parameters. Each variant is trained on 120B tokens with a global batch size of 1024, and comparisons use the same 30k-step window. The learning rate peaks at 2.54×10^{-3} and follows cosine decay to 3×10^{-5}. Auxiliary loss, DeepSeek's loss-free method, Quantile Balancing, and ID Balancing use the same architecture, data, and optimizer settings within each routing configuration.

Controller settings. DeepSeek's loss-free method uses a bias-update step size of $\eta = 0.001$, following the DeepSeek-V3 technical report (Liu et al., 2024). The Auxiliary loss baseline uses a balancing coefficient of $\alpha = 0.05$, except in the coefficient sweep in Appendix B.2. The main ID Balancing runs use $K_i = K_d = 6 \times 10^{-3}$. Quantile Balancing uses no EMA except in the diagnostic in Fig. 5. The integral-gain ablation fixes $K_d = 0$ and varies $K_i \in \{3, 6, 9\} \times 10^{-3}$. The derivative-gain ablation fixes $K_i = 6 \times 10^{-3}$ and varies $K_d \in \{0, 3, 6, 12\} \times 10^{-3}$. Both ablations use the Top-3-of-768 model over 30k steps. Appendix B.6.1 and Appendix B.6.2 give the corresponding trajectory analyses.

Algorithm 1 ID Balancing bias update, one MoE layer, per training iteration.

Require: expert count E, active experts K, integral gain K_i, derivative gain K_d

$b_i \leftarrow 0,\ e_i^{\text{prev}} \leftarrow 0$ for all $i \in [E]$ ▷ bias and previous error
for each training iteration **do**
  route the batch by $\mathcal{T}(\mathbf{x}) = \text{TopK}_{i\in[E]}(s_i + b_i)$ ▷ Eq. 2; bias frozen during the step
  $n_i \leftarrow$ number of tokens assigned to expert i ▷ summed over microbatches and the data-parallel group
  $\bar{n} \leftarrow \frac{1}{E}\sum_{i=1}^{E} n_i$ ▷ uniform target load
  for each expert $i \in [E]$ **do**
    $e_i \leftarrow (\bar{n} - n_i)/\bar{n}$ ▷ normalized load error, Eq. 3
    $\Delta e_i \leftarrow e_i - e_i^{\text{prev}}$
    $g_i \leftarrow \mathbb{1}[\, e_i^{\text{prev}}\, \Delta e_i > 0\,]$ ▷ 1 iff the imbalance is worsening (e_i^{prev} not yet updated)
    $b_i \leftarrow b_i + K_i\, e_i + K_d\, g_i\, \Delta e_i$ ▷ accumulate integral + gated derivative
    $e_i^{\text{prev}} \leftarrow e_i$
  end for
  $b_i \leftarrow b_i - \frac{1}{E}\sum_{j=1}^{E} b_j$ for all $i \in [E]$ ▷ center to zero mean
end for

Continued pretraining. We compare full ID gains ($K_i = K_d = 6 \times 10^{-3}$), reduced gains ($K_i = K_d = 6 \times 10^{-6}$), and frozen biases in the 28-layer Top-8-of-256 and Top-10-of-768 models. The learning rate is constant at 3×10^{-5}. ID Balancing uses the reduced gains for continued pretraining, while the DeepSeek and Quantile Balancing baselines freeze their biases during this stage. Downstream evaluation uses models trained on 560B tokens in total. Appendix B.1 discusses the gain comparison.

Model-scale comparison. Fig. 1(c) compares Top-10-of-768 models with 18.9B and 69.9B total parameters (1.03B and 3.2B active). Worst backbone MaxVio is computed over the first 30k and 50k steps of the small and large models, respectively. This comparison varies the complete model configuration, including depth, width, active parameters, and training duration, rather than isolating model size alone.

A.3 Metrics and Aggregation

MaxVio and MinVio are defined in Eq. 12. They measure the largest relative overload and underload among the routed experts of a layer at a given step. MaxVio is bounded by $E/K - 1$, while MinVio lies in $[0, 1]$ and reaches 1 when at least one expert receives no tokens. Lower values indicate better load balance. Last1k and Avg. denote means over the final 1k steps and the full comparison window, respectively. Backbone means average over the backbone layers, whereas Worst MaxVio takes the maximum across both layers and steps. The small models have 20 backbone layers, and the medium and large models have 28. MTP load metrics are reported separately. LM loss is averaged over the final 1k steps. In the quantitative tables, darker green denotes lower LM loss, darker red denotes larger load violations, and bold identifies the best value within each comparison block.

A.4 Downstream Benchmarks and References

We evaluate downstream performance on nine benchmarks. MMLU (5-shot) (Hendrycks et al., 2021a), MMLU-Pro (5-shot, Chain of Thought) (Wang et al., 2024c), and SuperGPQA (5-shot, Chain of Thought) (Du et al., 2026) assess general knowledge. MATH (4-shot, Chain of Thought) (Hendrycks et al., 2021b) and GSM8K (4-shot, Chain of Thought) (Cobbe et al., 2021) assess mathematical problem solving, while BBH (3-shot, Chain of Thought) (Suzgun et al., 2023) evaluates reasoning. MMMLU (5-shot) (OpenAI, 2024) evaluates multilingual understanding. EvalPlus (0-shot; the average over HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021), HumanEval+ and MBPP+) and MultiPL-E (0-shot; Python, C++, Java, PHP, TypeScript, C#, Bash, JavaScript) (Cassano et al., 2023) evaluate code generation.

Avg. in Tab. 2 is the arithmetic mean of the nine reported benchmark scores. The same benchmark suite is used for the prefill-mode expert-utilization analysis in Appendix B.5.

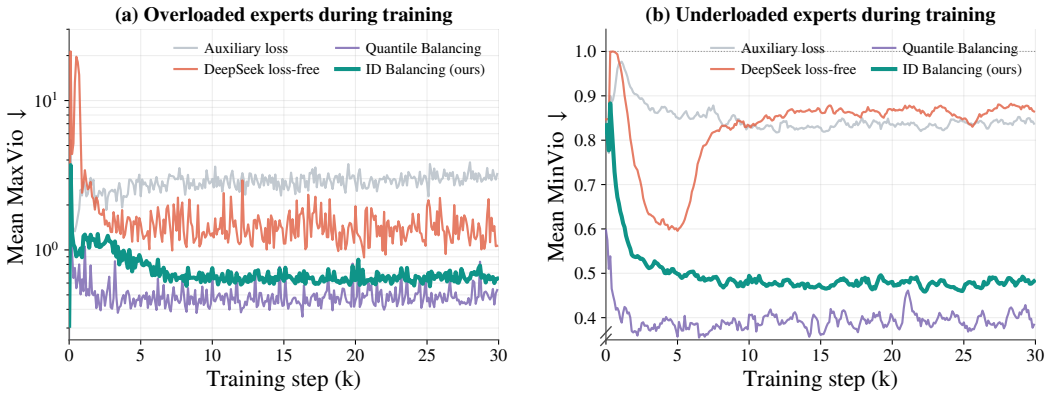


Figure A: **ID Balancing and Quantile Balancing maintain lower backbone violations at a higher learning rate.** Results use Top-10-of-768 routing and 120B training tokens at a constant learning rate of 5.86×10^{-3} ($2.3\times$ the standard peak), averaged over 20 backbone layers. **(a)** Mean MaxVio on a logarithmic scale (↓). **(b)** Mean MinVio (↓).

A.5 Bias-update procedure

Algorithm 1 gives the per-iteration update for one MoE layer, including initialization and token-count aggregation.

B Additional Experimental Details

B.1 Continued-pretraining load control

Fig. 6 compares full gains, reduced gains, and frozen biases under the settings in Appendix A.2. Its top row uses Top-8-of-256 routing and its bottom row uses Top-10-of-768 routing. Load metrics average over 28 backbone layers, and the plotted LM loss uses 201-step smoothing.

Freezing the bias removes active load correction as the data distribution changes. With Top-8-of-256 routing, the frozen baseline reaches mean MaxVio of 2.2–2.5 and MinVio of about 0.83. With Top-10-of-768 routing, mean MaxVio remains between 6 and 7, and MinVio reaches about 0.95. Full gains yield the highest LM loss in the comparison. Reducing both gains by $1000\times$ keeps LM loss within 0.001 of the frozen baseline during the second half of continued pretraining, while mean MaxVio settles near 1 and 1.4 in the two settings, respectively. These results support retaining active correction with smaller gains during continued pretraining.

B.2 Higher-learning-rate load control

We train the 1.03B-active/18.9B-total Top-10-of-768 model on 120B tokens at a constant learning rate of 5.86×10^{-3}, or $2.3\times$ the standard peak. Tab. 3 and Fig. A show lower backbone violations for ID Balancing and Quantile Balancing than for Auxiliary loss and DeepSeek's loss-free method. Their Last1k backbone MaxVio values are 0.6437 and 0.5021, compared with 3.1383 and 1.3755 for the two baselines. Quantile Balancing gives the strongest backbone balance, while ID Balancing has the lowest LM loss of 2.0007 and Last1k and Avg. MTP MaxVio of 0.4642 and 0.5068.

Increasing the auxiliary-loss coefficient does not restore balance in the tested range. Fig. B compares $\alpha \in \{0.05, 0.10, 0.50\}$. Larger coefficients yield higher LM loss, reaching about 2.45 for $\alpha = 0.50$, compared with 2.01 for $\alpha = 0.05$. At $\alpha = 0.50$, mean MaxVio exceeds 50. Both larger-coefficient runs reach mean MinVio of 1.0 and terminate before 30k steps.

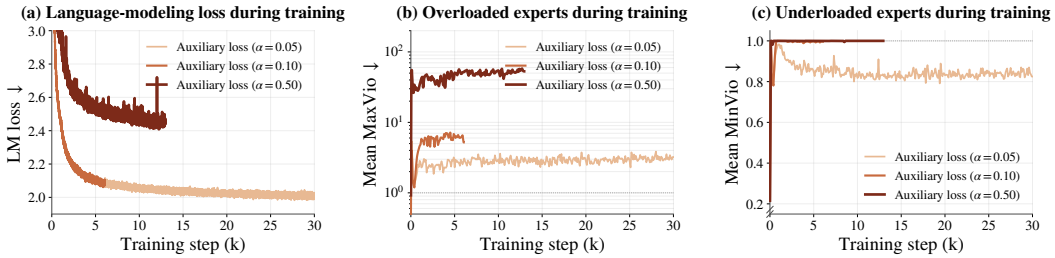


Figure B: **Larger auxiliary-loss coefficients worsen LM loss and load balance.** We compare $\alpha \in \{0.05, 0.10, 0.50\}$ in a 1.03B-active/18.9B-total model with Top-10-of-768 routing at a constant learning rate of 5.86×10^{-3} ($2.3\times$ the standard peak). Training targets 120B tokens, but both larger-α runs terminate before 30k steps. Load metrics average over 20 backbone layers. **(a)** LM loss (↓). **(b)** Mean MaxVio on a logarithmic scale (↓). **(c)** Mean MinVio (↓).

B.3 Layer-wise load balance

Fig. C complements Tab. 1 with MaxVio and MinVio for all 20 backbone layers under Top-3-of-768 routing. Auxiliary loss varies substantially across layers, with modest improvement in the final 1k steps. DeepSeek's loss-free method has high training-average overload in deeper layers, reaching MaxVio of about 6.0 at layer 19, compared with 0.9 in the final 1k steps. MinVio nevertheless remains near 1.0 in several layers.

Quantile Balancing and ID Balancing show more consistent layer-wise profiles and smaller differences between training-average and final-stage values. ID Balancing has lower or comparable MaxVio and lower MinVio than Quantile Balancing across all 20 layers. MinVio still generally increases with depth for all four methods.

B.4 Gradient and activation dynamics

Fig. D tracks global gradient norms and MoE-output magnitudes in the 3.0B-active/24.8B-total Top-8-of-256 model, with output magnitudes averaged over 28 backbone layers (Wang et al., 2024a; Xiong et al., 2020). ID Balancing and Quantile Balancing end with gradient norms near 0.043, below Auxiliary loss at 0.050 and DeepSeek's loss-free method at 0.049. DeepSeek's method also shows stronger late-stage fluctuations. Output magnitudes rise early and then decline, ending near 0.54 for Quantile Balancing and 0.51–0.52 for the other methods.

B.5 Expert utilization at inference

We record layer-wise expert loads and router scores from continued-pretrained models in prefill mode on nine downstream benchmarks, averaging each layer value over the benchmarks. Models use Top-8-of-256 or Top-10-of-768 routing and share the same architecture within each setting. ID Balancing uses $K_i = K_d = 6 \times 10^{-6}$ during continued pretraining. The benchmark suite and its references are described in Appendix A.4.

Fig. E(a, d) reports the *inactive-expert ratio*, the fraction of experts receiving no evaluation token. Under Top-8-of-256 routing, ID Balancing has an average ratio of 2.8%, compared with 2.9% for DeepSeek's loss-free method, 3.3% for Quantile Balancing, and 4.7% for Auxiliary loss. Under Top-10-of-768 routing, it reduces the average from 8.4% to 6.4% and the first-layer ratio from 9.3% to 3.4% relative to Auxiliary loss. These results are consistent with the lower training-time MinVio in Fig. C. In the Top-8-of-256 setting, inactive ratios generally increase with depth across methods.

Fig. E(b, e) reports top-K gating-score entropy, $H_{\text{top-}K} = -\sum_{i=1}^{K} w_i \log w_i$ in nats, where w_i is the normalized combination weight of selected expert i. Under Top-8-of-256 routing, the loss-free methods have higher entropy than Auxiliary loss, indicating more even mixtures, with small differences among the three loss-free methods.

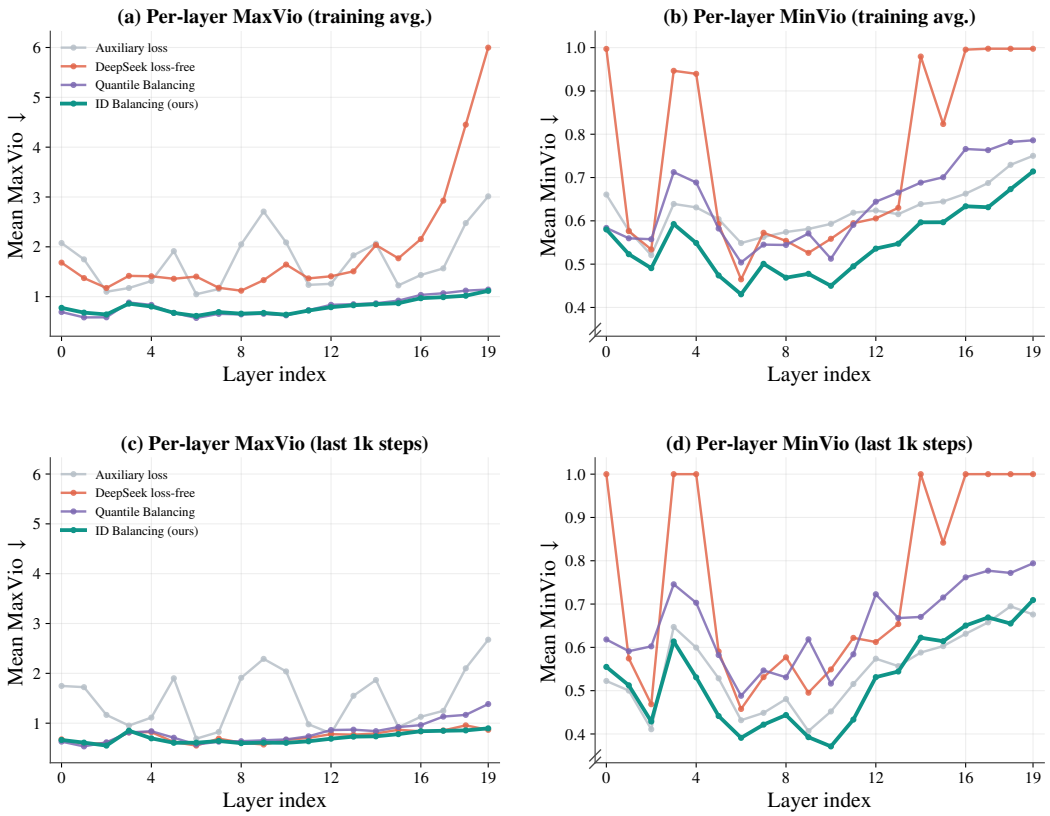


Figure C: **ID Balancing maintains consistent layer-wise load control.** Results cover 20 backbone layers under Top-3-of-768 routing over 30k steps. The top and bottom rows show training and final-1k-step averages, respectively. **(a, c)** MaxVio (↓). **(b, d)** MinVio (↓).

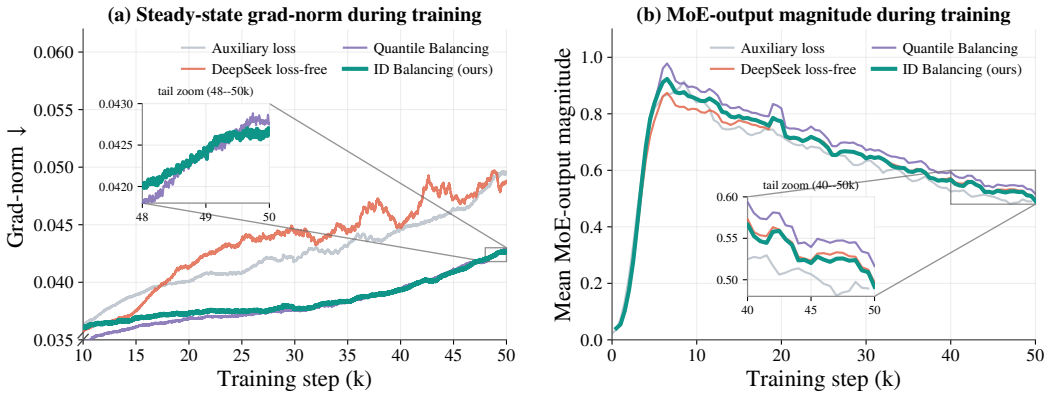


Figure D: **Load-balancing methods show distinct gradient and activation dynamics.** Results use a 3.0B-active/24.8B-total model with Top-8-of-256 routing over 50k steps. **(a)** Global gradient norm. **(b)** MoE-output magnitude averaged over 28 backbone layers.

Fig. E(c, f) reports full-pool selection-score entropy, $H_{\text{full}} = -\sum_{e=1}^{E} p_e \log p_e$. For Auxiliary loss, p_e is the softmax probability over router logits. For loss-free methods, it normalizes the non-negative parts of $\sigma(z_e) + b_e$ across experts. Higher entropy indicates a flatter distribution, but absolute values are not directly comparable across these score families. Under Top-8-of-256 routing, loss-free score distributions sharpen with depth, while Auxiliary loss shows the opposite trend.

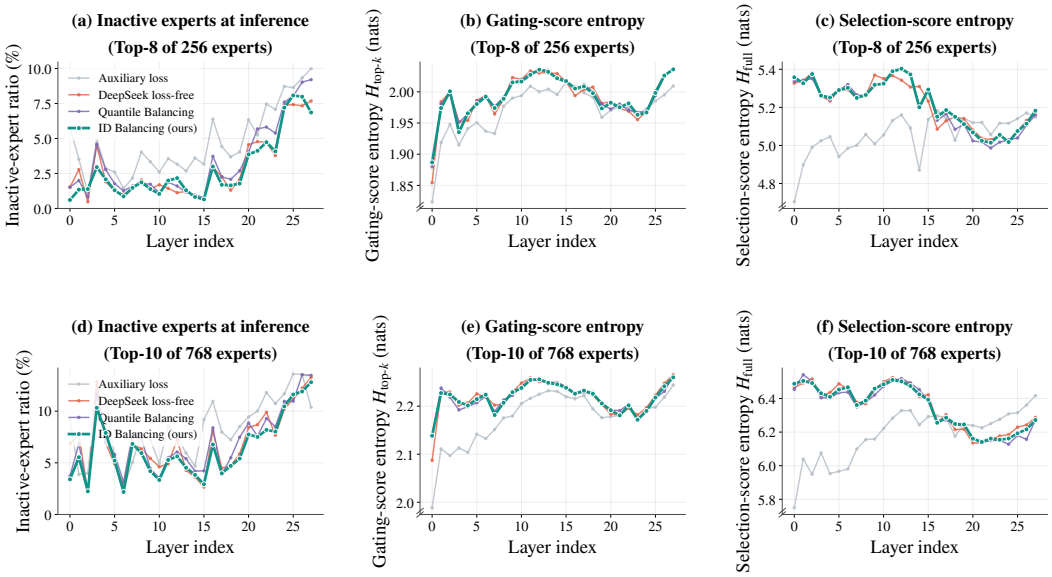


Figure E: **ID Balancing reduces expert inactivity relative to Auxiliary loss.** Layer-wise results average over nine downstream benchmarks. The top and bottom rows use Top-8-of-256 and Top-10-of-768 routing. **(a, d)** Inactive-expert ratio: the fraction of experts receiving no evaluation token (↓). **(b, e)** Top-K gating-score entropy: higher values indicate a more even mixture over selected experts. **(c, f)** Full-pool selection-score entropy: higher values indicate a flatter score distribution.

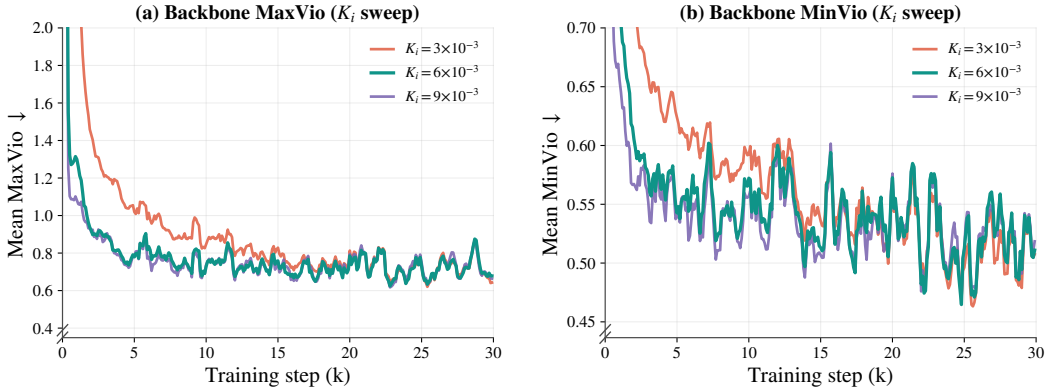


Figure F: **A small integral gain slows early backbone load correction.** Results use Top-3-of-768 routing over 30k steps with $K_d = 0$ and $K_i \in \{3, 6, 9\} \times 10^{-3}$. **(a)** Mean backbone MaxVio (↓). **(b)** Mean backbone MinVio (↓).

B.6 Gain-ablation trajectories

B.6.1 Integral gain

With $K_d = 0$, we vary $K_i \in \{3, 6, 9\} \times 10^{-3}$. The smallest gain corrects early backbone imbalance more slowly, while later trajectories are similar (Fig. F). Increasing K_i from 3×10^{-3} to 6×10^{-3} reduces training-average backbone MaxVio from 0.9459 to 0.8003 and worst-case MaxVio from 23.26 to 14.96 (Tab. 4). At 9×10^{-3}, the average falls slightly further to 0.7872, but worst-case backbone MaxVio rises to 20.13 and worst-case MTP MaxVio rises from 15.18 to 35.56. Although the smallest gain retains the lowest LM loss and several final-stage and MTP metrics, we choose $K_i = 6 \times 10^{-3}$ for stronger early correction and the lowest worst-case backbone MaxVio among the tested gains.

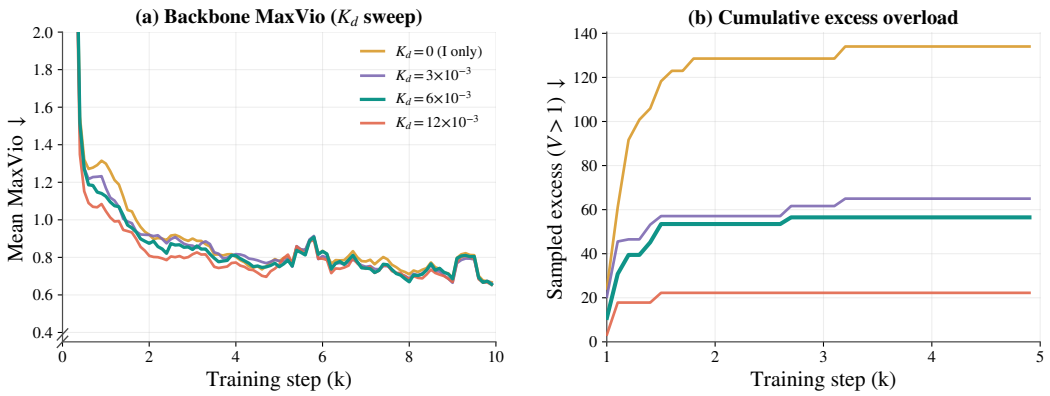


Figure G: **The derivative term reduces early backbone overload.** Results use Top-3-of-768 routing with $K_i = 6 \times 10^{-3}$ and $K_d \in \{0, 3, 6, 12\} \times 10^{-3}$, where $K_d = 0$ is the integral-only baseline. Let V_t be mean MaxVio over 20 backbone layers. **(a)** First 10k steps with five-point smoothing. **(b)** Cumulative excess above $V_t = 1$ over 1–5k steps.

B.6.2 Derivative gain and cumulative excess overload

With $K_i = 6 \times 10^{-3}$ fixed, we vary $K_d \in \{0, 3, 6, 12\} \times 10^{-3}$. Adding the derivative term improves early backbone control, while later trajectories remain similar (Fig. G(a)). Relative to I only ($K_d = 0$), the default $K_d = 6 \times 10^{-3}$ reduces mean backbone MaxVio from 2.0020 to 1.9330 over 0–1k steps and from 0.9059 to 0.8612 over 1–5k steps, with lower MinVio in both intervals (Tab. 5). Over 1–5k steps, its cumulative excess above layer-mean MaxVio $V_t = 1$ is 57.9% lower than I only, using unsmoothed observations (Fig. G(b)).

The full-training backbone improvement is modest, from 0.8003 to 0.7931 in Avg. MaxVio, while MTP Avg. MaxVio rises from 0.8862 to 0.9698. We choose $K_d = 6 \times 10^{-3}$ as a compromise: it gives stronger backbone control over 1–5k steps than 3×10^{-3}, with less MTP overload than 12×10^{-3}. It also yields the lowest LM loss and training-average backbone MinVio among the tested gains.

C Limitations

Our evaluation covers several routing sparsities, model sizes, and training conditions within a family of decoder-only MoE models using fixed Top-K routing. Extending the evaluation to other architectures and routing methods is a natural next step. The accumulated worsening-gated derivative update is supported empirically. The scaling experiments compare complete model configurations that differ in depth, width, active and total parameters, and training duration, so they assess load control across configurations without isolating the effect of model size alone.