An Empirical Study on Noisy Data and LLM Pretraining Loss Divergence

Qizhen Zhang[2,†] **Ankush Garg**[†] **Jakob Foerster**[1,2]
Niladri Chatterji[†,*] **Kshitiz Malik**[†,*] **Mike Lewis**[1,*]
[1]Meta Superintelligence Labs [2]University of Oxford

Abstract

Large-scale pretraining datasets drive the success of large language models (LLMs). However, these web-scale corpora inevitably contain large amounts of noisy data due to unregulated web content, or randomness inherent in data. Although LLM pretrainers often speculate that such noise contributes to instabilities in large-scale LLM pretraining and, in the worst cases, loss divergence, this phenomenon remains poorly understood. In this work, we present a systematic empirical study of whether noisy data causes LLM pretraining divergences and how it does so. By injecting controlled synthetic uniformly random noise into otherwise clean datasets, we analyze training dynamics across model sizes ranging from 480M to 5.2B parameters. We show that noisy data indeed induces training loss divergence, and that the probability of divergence depends strongly on the noise type, amount of noise, and model scale. We further find that noise-induced divergences exhibit activation patterns distinct from those caused by high learning rates (Wortsman et al., 2023), and we provide diagnostics that differentiate these two failure modes. Together, these results provide one of the first large-scale, controlled characterizations of how noisy data affects loss divergence in LLM pretraining.

1 Introduction

Scaling up transformer models in terms of model size and datasets has led to remarkable capabilities, but the resulting large-scale pretraining runs are extremely expensive (Dubey et al., 2024; Adcock et al., 2026). Furthermore, not every pretraining run succeeds (Chowdhery et al., 2023; Zhang et al., 2022): researchers frequently observe instabilities that slow or disrupt learning, and in the worst case, the pretraining loss diverges entirely. Understanding these failures is therefore essential.

One hypothesized cause of loss divergence is noisy data in the training corpus. Web-scale datasets inevitably contain substantial noise due to unregulated web content or randomness inherent in data. Yet whether such noise truly drives divergence remains poorly understood.

In this work, we present a systematic empirical study of how uniform random noisy data impacts LLM pretraining stability. We focus on the worst-case instabilities that lead to loss *divergence* rather than fast loss spikes. By injecting controlled synthetic uniform random noise into otherwise clean datasets, we examine training dynamics across model sizes from 480M to 5.2B parameters. Our main contributions are fourfold:

1. **Noise can cause divergence & the type of noise matters** (§ 4.1): We show that injected noise can indeed cause pretraining loss divergence. We also find that different noise types affect stability to varying degrees, and that some forms of noise cause divergence even in $<$ 1B parameter models, which are typically very stable to train.
2. **Scaling trends of noisy data** (§ 4.2): We show that higher data noise ratios increase the probability of loss divergence, and that larger models, particularly deeper models rather than wider models, are substantially more likely to diverge.

[*] Joint last author, [†] Work done at Meta
Correspondence: qizhen.zhang@eng.ox.ac.uk

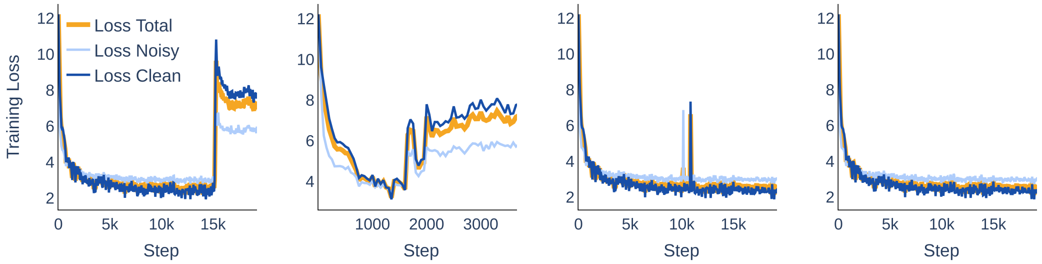

Figure 1: Examples of four pretraining runs using the same 1.3B model architecture and 15% noisy data, differing only in random seed. The left two runs illustrate cases where training diverges, while the right two show stable behavior. We focus on divergences rather than fast spikes, so the third run is categorized as stable because its loss spikes quickly recover and the loss continues to decrease.

3. **Noisy data divergence differs from high learning rate divergence** (§ 4.3): We show that loss divergences induced by noisy data exhibit activation patterns that differ from those caused by overly high learning rates, allowing clear diagnostic separation between the two failure modes.
4. **Dense vs. MoE sensitivity to noisy data** (§ 4.5): Finally, we show that dense models and active parameter matched Mixture-of-Experts (MoE) models have comparable sensitivity to noisy data.

Together, these results provide one of the first large-scale, controlled characterizations of how noisy data causes LLM pretraining loss divergence, which we hope will encourage future research to better understand and address the mechanisms through how data quality causes training instability.

2 Related work

LLM pretraining instabilities Many studies analyze pretraining instabilities through the lens of abnormal activation and parameter behavior. Approaches such as QK-layer normalization (Dehghani et al., 2023), z-loss regularization (Chowdhery et al., 2023), and weight decay (Loshchilov & Hutter, 2017) stabilize training by modifying model architectures or loss functions to mitigate these abnormal behaviors. In contrast, relatively few works examine how *training recipe decisions* give rise to these activation and parameter pathologies in the first place. Prior research in this direction has examined numerical precision and quantization, showing that reduced precision can induce pathological activation and gradient behavior (Micikevicius et al., 2017; Fishman et al., 2024; Peng et al., 2023), as well as learning rate choices, where Wortsman et al. (2023) demonstrate that overly high learning rates can cause abnormal activation and parameter growth and lead to loss divergence even in small language models. Our work studies an orthogonal training recipe choice: data quality. We show that noisy pretraining data can cause loss divergence, and that noisy data-induced failures exhibit activation and parameter patterns distinct from those caused by high learning rates. Based on this distinction, we introduce diagnostic methods that separate learning rate induced failures from noisy data induced failures, enabling practitioners to identify when data cleaning is required.

LLM pretraining data quality The most closely related work is Ru et al. (2025), which argues that noisy data has limited impact on training dynamics, reporting only modest loss increases and no training spikes or divergence. However, their analysis considers a narrow noise setting: injecting clean text with tokens sampled from the full tokenizer vocabulary. We show that noise type is important: certain forms of random noise, especially when restricted to a *subset* of the tokenizer vocabulary, significantly increase the probability of loss divergence. By systematically varying noise types, noise ratios, and model sizes, we demonstrate that noisy data can destabilize training, and we characterize when this instability arises and how it can be diagnosed.

Noisy label works prior to LLMs A substantial body of theoretical and empirical research has studied neural network training under noisy labels in supervised learning settings (Zhang & Sabuncu, 2018; Chen et al., 2019; Ghosh et al., 2017; Sukhbaatar et al., 2014; Chen et al., 2023; Zhou et al., 2019; Rolnick et al., 2017; Patrini et al., 2017; Natarajan et al., 2013). These works typically focus on small-scale models and datasets, studying how noisy labels affect model performance, generalization, and overfitting. In contrast, our work studies *large-scale*, *transformer-based* language models and investigates how noisy data leads to *training loss divergence*. As demonstrated in § 4.2, these instabilities become increasingly pronounced at larger model scales: configurations that are stable for small models (where most prior noisy-label studies are conducted) do not remain stable when scaled to the regimes we consider. Moreover, we provide diagnostic insights for identifying noisy data induced failures specific to transformer architectures.

3 Experimental methodology

3.1 Noisy data generation

Our study focuses on *uniform random noise*, which can arise from unregulated web content or inherently random sequences such as hash codes. Collecting a sufficiently large corpus of genuine random noise from web crawls is computationally infeasible. Instead, we simulate uniform random noise synthetically in a controlled setting.

We construct noisy training data from a clean corpus D_c, consisting of a subset of the Llama 4 pretraining data mixture (Adcock et al., 2026). Let V denote the tokenizer vocabulary and let $\alpha \in (0, 1)$ be the target noise ratio, the percentage of tokens that are artificial noise. Each noise token is sampled independently from a designated noise vocabulary, $n_i \sim \text{Uniform}(V_N)$, where $V_N \subseteq V$. We introduce a restricted noise vocabulary because real-world noise typically occupies a limited subset of the tokenizer vocabulary, for example, hash-like strings are composed of digits and lowercase letters (e.g., `4f2a9c1e0d`). We defer details on the choice of V_N to § 4.1. For each document with n tokens, we inject uniform random noise using one of the following two approaches. **Inserting:** We sample $\frac{n\alpha}{1-\alpha}$ insertion positions. At each sampled position i, we insert a noise token n_i between the i-th and $i+1$-th clean tokens. Positions are sampled with replacement, allowing consecutive noise tokens. **Overwriting:** Each token in the document is replaced by a noise token with probability α.

3.2 Dense model architecture details

We train decoder-only transformer models (Vaswani et al., 2017) using the standard auto-regressive next-token prediction objective. Our dense architecture follows the Llama 3 family of models (Dubey et al., 2024): we use pre-normalization transformers, we do not use QK-layernorm (Dehghani et al., 2023), we do not use biases, we do not tie the input and output embedding weights, and we use rotary positional embeddings (Su et al., 2024) and group query attention (Ainslie et al., 2023). Table 1 summarizes the shared hyperparameters across all models.

To vary model size, we adjust the number of layers and the model dimension which are detailed in § 4.2 and Table 2. We always jointly scale up the model dimension and number of query heads. Reported parameter counts include all parameters, including input and output embeddings.

3.3 MoE architecture details

For Mixture of Experts (MoE; Shazeer et al., 2017; Fedus et al., 2022) experiments, we use dropless MoEs (Gale et al., 2023; Liu et al., 2024) with 16 feed-forward network (FFN) experts and token-choice top-2 routing (see § A.1 for more details). For fair comparison with dense models, we active-parameter match top-2 MoE models by scaling the FFN dimension by 0.5. All other architectural components remains the same as the dense setting.

3.4 Training details

We use AdamW (Loshchilov & Hutter, 2017) with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 1e-8$, global norm gradient clipping of 1.0, and weight decay $1\text{e}-4$. We apply a learning rate (LR) schedule with a 2000-step linear warm-up followed by cosine decay, with a minimum LR ratio of 0.1 and 2e4 total

Example: `https://docs.oracle.com/cd//E19528-01//820-0890//WebSvr.html#wp35099`

Example: `https://cdn.kernel.org/pub/linux/kernel/v6.x/ChangeLog-6.16?`

steps. We use a peak LR of $lr = 1.85e - 2$ and a batch size of $2.6\text{e}5$ tokens, and we train for less than one epoch in all experiments. To enable hyperparameter transfer across model sizes, we adopt Maximal Update Parametrization (Yang et al., 2021) with a base width of 256. All experiments are trained on 16 to 64 H100 GPUs using PyTorch FSDP2 (Paszke et al., 2019).

3.5 Measuring training stability

Training stability is inherently non-deterministic. Even with identical architectures and training setups, runs that differ only by random seed can behave differently. As shown in Figure 1, among four such runs, two diverge while two remain stable. Because this work focuses on loss *divergence*, we treat the third run as stable, since it quickly recovers from a loss spike and continues to decrease. We mark a run as diverged if its loss exceeds the minimum observed so far by more than 0.5 nats/token for at least 600 consecutive steps. To quantify training stability, we estimate the probability of divergence by repeating each experiment with 20 random seeds and reporting the percentage of runs that diverge. To ensure that any observed divergences are due to the injected noise, we verify that for all model sizes studied, none of the 20 seeds trained on the clean corpus D_c (i.e., without any noise injection) diverges when using the same training setup and hyperparameters.

For analyses comparing relative stability across settings (§ 4.1, § 4.2), we instead use an LR schedule designed for 15 trillion tokens, following Dubey et al. (2024), to approximate large-scale pretraining. Due to computation constraints, these runs are truncated at $2\text{e}4$ steps, and we consider only divergences occurring within this horizon.

4 Results and analysis

We first show that uniform random noise can indeed cause pretraining loss divergence and identify the most destabilizing noise types (§ 4.1). Using the most destabilizing noise type, we then study how divergence scales with noise ratio α and model size (§ 4.2). Next, we present diagnostics that distinguish divergences caused by high LRs, as described by Wortsman et al., from those induced by noisy data (§ 4.3). Finally, we show that dense and MoE models exhibit similar sensitivity to noisy data training conditions (§ 4.5).

4.1 Noisy data causes divergence & the type of noisy data matters

We first establish that noisy training data can induce loss divergence, and then study which types of the noise are most destabilizing. All experiments in this subsection use a noise ratio of $\alpha = 55\%$ and a 540M dense model following the architecture in Table 1 and Table 2, with model dimension 1024 and 10 layers.

Question: Can noisy data cause training divergence? If so, does the **size** of the noise vocabulary $|V_n|$ affect the probability of divergence?
Answer: Yes to both.

In Figure 2, we vary the noise vocabulary size $k = |V_n|$ by restricting noise tokens to the first $k \in 5, 50, 500, 5000, 50000$ tokens in the tokenizer vocabulary, and also consider using the full vocabulary ($|V_n| = |V| = 200{,}000$). These experiments use insertion noise. We defer discussion of alternative choices for the k tokens, as well as insertion versus overwriting noise, to later in this section. We see that injecting noise indeed induces loss divergence, with smaller noise vocabularies increasing the probability of divergence.

Question: Does the **content** of the noisy token vocabulary affect the probability of divergence?
Answer: No.

One may suspect that inserting common versus rare tokens as noise would affect training dynamics differently. To test this, we fix the noise vocabulary size to $|V_n| = 5$, and we vary its *content* by

In addition to different initialization, non-determinism from GPU execution may also be a cause.

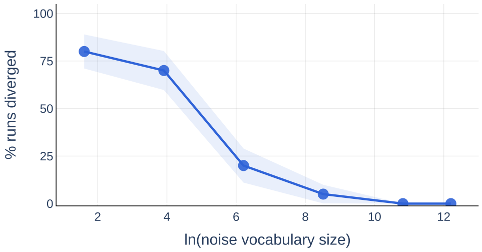

Figure 2: **Effect of noise vocabulary size on training stability** for 540M dense models with noise ratio $\alpha = 55\%$. We observe that reducing the size of the noise vocabulary significantly increases the probability of divergence. In all plots throughout the paper, we use the shaded regions denote to standard error.

selecting different sets of five tokens from the tokenizer vocabulary V. We construct token sets $t_1, \ldots, t_5 \subset V$ whose average frequency in the clean corpus D_c span several orders of magnitude, ranging from sets of extremely common tokens (each appearing on average $\sim$ 12 million times) to sets of tokens that never appear in D_c.

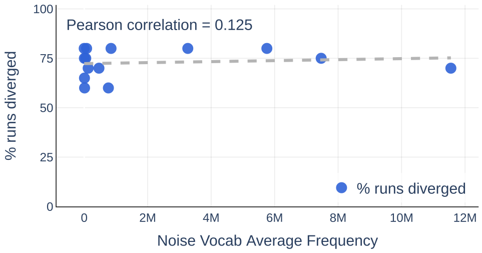

Figure 3: **Effect of noise vocabulary content on training stability.** Fixing the noise vocabulary size to $|V_n|$=5, we plot the divergence rate against the average frequency of the selected noise tokens in the clean corpus. We observe that the content of the noise vocabulary has little effect on loss divergence with a near zero Pearson correlation.

In Figure 3, the x-axis shows the average frequency of the five noise tokens in the clean dataset, $\frac{1}{5}\sum_{i=1}^{5} \text{frequency}(t_i, D_c)$, while the y-axis reports the percentage of runs that diverge. We find that the actual noisy token content has minimal effect on training stability, with a near-zero Pearson correlation of 0.125.

Question: Does **inserting vs overwriting** noise affect divergence in LLM pretraining?
Answer: Yes.

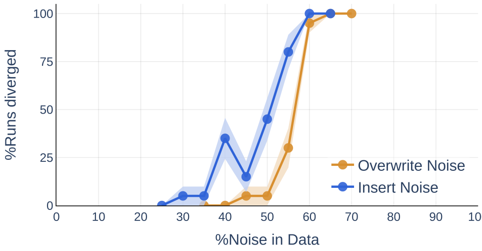

Figure 4: **Comparison of inserting versus overwriting noise** for a fixed noise vocabulary of five tokens. Inserting noise results in a higher probability of loss divergence than overwriting noise.

In Figure 4, we compare inserting versus overwriting noise, fixing the noise vocabulary to the first five tokens in the tokenizer vocabulary for both settings. We observe that inserting noise leads to higher probability of loss divergence than overwriting noise.

4.2 Scaling trends of noisy data

Next, we study how sensitivity to noisy data scales along two axes: model size and noise ratio. For all remaining experiments, we use the most destabilizing noise setting identified in § 4.1, namely inserting noisy tokens from a noise vocabulary of size $|V_n| = 5$.

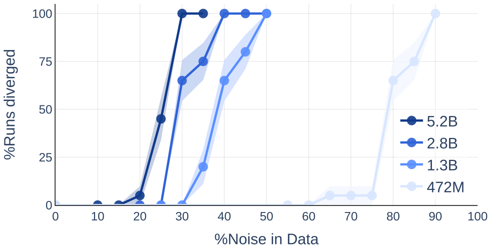

Figure 5: Scaling model size by jointly scaling the depth and width of the model. We observe larger models are more sensitive to noise, and higher noise ratio leads to more divergences.

In Figure 5, we scale up model size by jointly increasing the number of layers and the model dimension while maintaining a fixed ratio $\frac{\text{num\_layers}}{\text{model\_dim}} = 204.8$. We consider four dense models ranging from 472M parameters (1024 dimensions, 5 layers) to 5.2B parameters (4096 dimensions, 20 layers). We observe that i) at a fixed noise ratio, larger models diverge more frequently than smaller ones, and ii) increasing the noise ratio consistently raises the divergence rate across all model sizes.

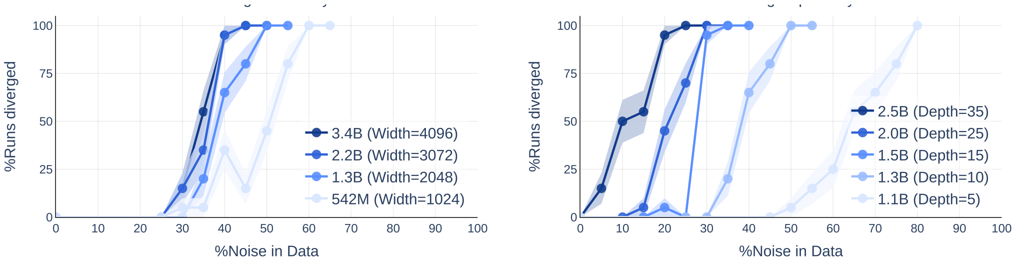

Figure 6: **Left:** Scaling model size by width only. **Right:** Scaling model size by depth only. We observe that increasing depth induces more divergences than increasing width, with deeper models diverging far more frequently.

In Figure 6, we isolate the effects of width and depth. The left panel fixes depth at 10 layers and scales width from 1024 to 4096, increasing model size from 542M to 3.4B parameters. The right panel fixes width at 2048 and scales depth from 5 to 35 layers, increasing model size from 1.1B to 2.5B parameters. Scaling width has a limited effect on training stability: despite spanning a larger parameter range, width-scaled models exhibit more similar divergence rates. In contrast, increasing depth substantially degrades stability. In the most extreme case, the 35-layer, 2.5B parameter model diverges in 15% of runs even at a noise ratio of 5%.

4.3 Divergences due to high LR vs. noisy data

Question: Can we distinguish divergences caused by high LRs from those caused by noisy data?
Answer: Yes, through examining activations.

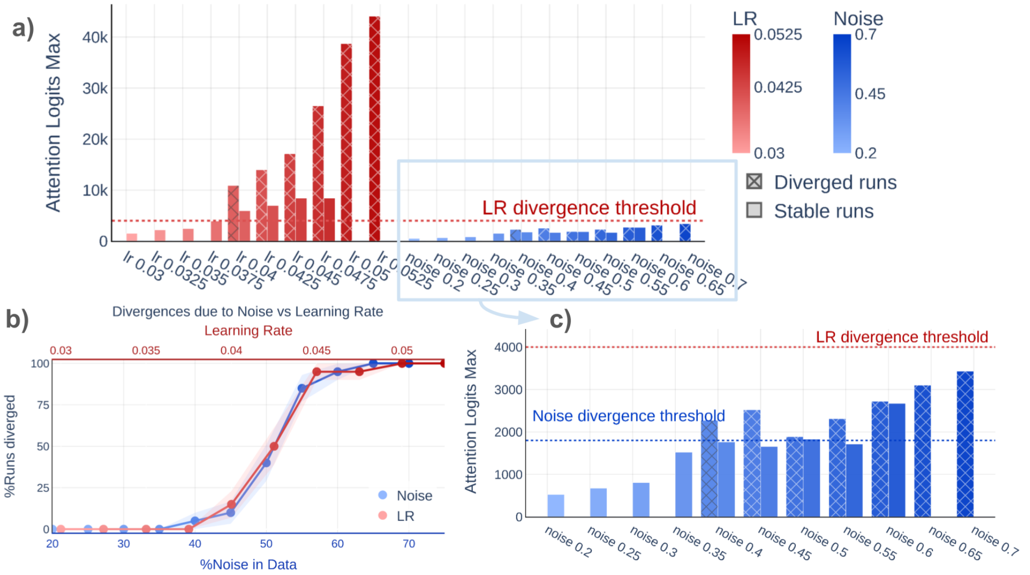

Figure 7: **Noisy-data and overly high LR divergences exhibit distinct activation behavior.** We compare 540M dense models trained on clean data with varying high-LRs (red) versus models trained on a small LR with varying noise ratios (blue). **(a)** Bars show the maximum attention logit at step 1000 averaged across seeds. Solid bars correspond to stable runs within the 20 seeds, while cross-hatched bars correspond to divergent runs within the 20 seeds. Missing bars indicate that no runs of that type occur for the configuration (i.e., lr=0.03 has no cross-hatched bars, meaning none of the 20 seeds for this configuration have diverged). Wortsman et al. identify divergence threshold for high-LR runs, shown by the dotted red line at 4000. **(b)** For comparability, LR and noise ratio ranges are selected to match divergence probabilities across the two settings. **(c)** A zoom-in on noisy-data runs from (a) reveals a different and lower divergence threshold for noisy data settings at approximately 1800 (blue dotted lines). As shown in Appendix B, the same numerical thresholds, ~ 4000 for high-LR runs and ~ 1800 for noisy-data runs, also holds across other model sizes.

4.3.1 Examining maximum attention logits

In a transformer's self-attention layer, attention logits are computed as $z_{ij} = \frac{\langle q_i, k_j \rangle}{\sqrt{d_h}}$. Dehghani et al. (2023) show that an excessively large maximum attention logit cause the softmax to collapse toward one-hot weights, destabilizing training. Building on this, Wortsman et al. (2023) find that overly high LRs drive this logit growth and lead to loss divergence. They also show that divergences can be predicted early in training when the maximum attention logit exceeds some threshold which holds for all model sizes.

In Figure 7, we analyze the maximum attention logits in runs that diverge due to noisy training data and compare them to runs that diverge due to high LRs using 540M dense models. See Appendix B for other model sizes including MoE models. For noisy-data runs, we train 20 seeds per noise ratio, with $\alpha \in [30\%, 70\%]$, using a fixed LR of 1.85×10^{-2}. For high-LR runs, we train 20 seeds per LR, with LR $\in [3.0 \times 10^{-2}, 5.25 \times 10^{-2}]$, on the clean dataset D_c. As shown in Figure 7(b), these ranges are chosen so that the probability of divergence under high LRs (red) closely matches that under noisy data (blue), enabling direct comparisons of activation and parameter statistics across the two settings. For each configuration, we report the maximum attention logit at step 1000, computed across seeds and reported separately for diverged runs (cross-hatched bars) and stable runs (solid bars). We chose step 1000 because the earliest divergence occurs at this point. We make two observations:

High LR divergences produce significantly larger maximum attention logits . In Figure 7(a), we first reproduce the findings of Wortsman et al. (2023): across model sizes (see Appendix B, including MoE architectures), high LR configurations lead to a non-zero probability of divergence once the maximum attention logit exceeds a threshold of 4000 at step 1000 . Importantly, this threshold should be interpreted at the *configuration level*: if runs within a configuration exceed this value, that configuration is unstable, i.e., at least one of the 20 seeds diverges. Within such unstable configurations, stable runs may or may not exceed the threshold, but all diverged runs consistently do. Beyond reproducing this result, we find that high-LR runs have significantly larger maximum attention logits than noisy-data runs; even at the highest noise ratios where noisy runs diverge 100% of times, the noisy-data runs' maximum attention logits never exceed the LR divergence threshold of 4000.

Noisy-data divergences occur at their own threshold, around 1800. In Figure 7(c), we see that within the noisy-data setting, configurations whose maximum attention logits exceed ~ 1800 exhibit a non-zero probability of divergence. This threshold is clearly separated from that of high-LR runs. As shown in Appendix B, the same numerical thresholds, ~ 4000 for high-LR runs and ~ 1800 for noisy-data runs, also apply across other model sizes including MoE models.

Together, these results show that divergence from noisy data is mechanistically distinct from divergence induced by high LRs. Maximum attention logits also provide a simple diagnostic for identifying the underlying cause of divergence. If a run diverges without ever exceeding the high-LR divergence threshold (~ 4000) at step 1000 but exceeds the noisy-data threshold (~ 1800), then the divergence is likely attributable to noisy data rather than overly high LR.

4.3.2 Examining parameter norms and other activations

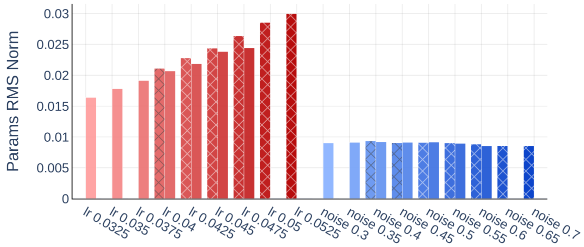

Figure 8: Parameter RMS norms for 540M dense models trained on clean data with varying LRs (red) versus on varying noisy data with a fixed small LR (blue). Divergences induced by noisy data exhibit significantly smaller parameter norms than those induced by high LRs. Increasing the LR also leads to systematic growth in parameter norms, whereas increasing the noise ratio does not.

Beyond differences in maximum attention logits, we observe additional distinctions between divergences induced by high LRs and those caused by noisy data. As shown in Figure 8, even at comparable divergence rates, runs that diverge due to noisy data exhibit smaller parameter norms than those diverging due to high LRs. Moreover, increasing the LR leads to larger parameter norms, whereas increasing the noise ratio does not. See Appendix B for analysis on other activation statistics.

4.4 Interventions for noisy data

Question: If we diagnose a run as diverging due to noisy data, how can we stabilize training?
Answer: Cleaning the data is most effective; when this is infeasible, apply QK-layernorm.

We train a 540M dense model with noise ratios ranging from 30% to 70%. At 30% noise, no runs diverge, while at 70% noise all runs diverge across 20 seeds; intermediate noise levels yield intermediate divergence rates (see Figure 7b). In Figure 9, we report the average loss of only runs that remain stable across the 20 seeds. We show the loss on clean tokens only, since this reflects

In Wortsman et al. (2023), the reported threshold is 10^4 at step 2000. The lower threshold here is expected, as we measure activations earlier due to a shorter warm-up and fewer total training steps.

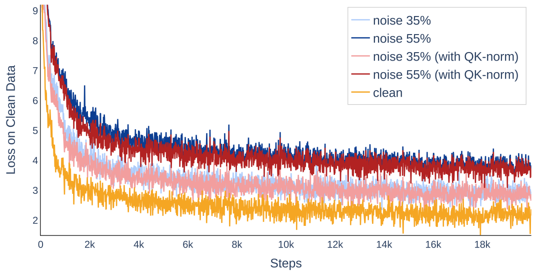

Figure 9: Average training loss for stable runs of a 540M-parameter dense model, evaluated on clean tokens only. Results are averaged over seeds that do not diverge.

the test-time distribution. For clarity, we only show results for $\alpha = 0\%$ (clean data), 35%, and 55% noise, although the trend is consistent across all noise ratios. We make two observations:

Cleaning data is the best solution. Even when some runs remain stable under noisy training data settings, increasing the noise ratio consistently degrades performance on clean tokens. Training on clean data yields the best training loss.

When data cleaning is not possible, use QK-layernorm. Applying QK-layernorm substantially reduces maximum attention logits for all noisy runs (see Figure 23) and eliminates all divergences across all noise ratios and all seeds. Even at 70% noise, a configuration that exhibited 100% divergence, QK-layernorm fully stabilizes training across all seeds. As shown in Figure 9, when trained on noisy data, QK-layernorm also yields slightly lower training loss compared to stable runs without QK-layernorm.

4.5 Dense vs. MoE models

Prior work shows that MoE architectures can be less stable than dense models, often requiring additional stabilization techniques (Zoph et al., 2022; Fedus et al., 2022). With noisy data, a natural concern is whether expert specialization amplifies this instability. In particular, one can question if some experts may disproportionately receive noisy tokens, effectively behaving like more unstable sub-network trained on highly noisy data.

Question: Do MoEs diverge more easily than dense models under noisy data?
Answer: No.

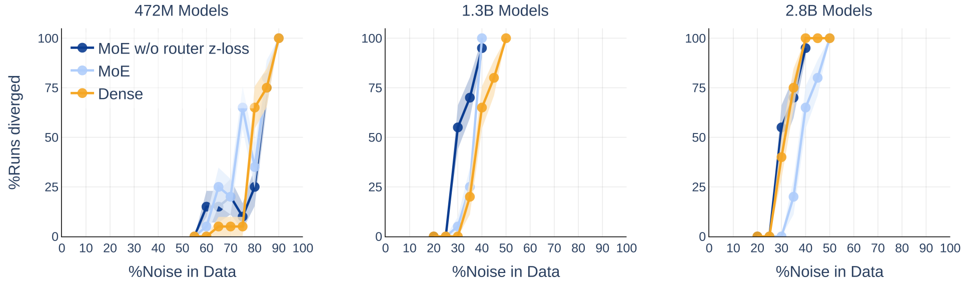

Figure 10: Divergence probability due to noisy data for dense and active-parameter–matched MoE models at three scales (472M, 1.3B, and 2.8B parameters). For each scale, we report results for MoEs trained with and without router z-loss regularization. MoE models diverge at similar rates to their dense counterparts across all scales.

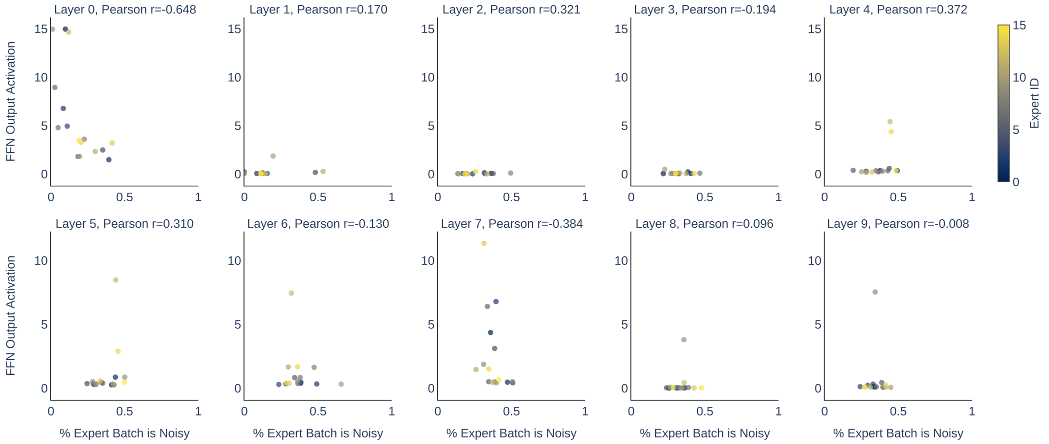

Figure 11: Each subplot corresponds to one layer in the 10 layer 1.3B MoE model. The x-axis shows the fraction of noisy tokens in the expert batch, and the y-axis shows the absolute mean of the FFN output activation. The average Pearson correlation across layers is -0.009, indicating little correlation between the two.

We compare dense models with active-parameter–matched MoE models at three scales (472M, 1.3B, and 2.8B parameters). We evaluate MoEs both with and without router z-loss regularization (Zoph et al., 2022), which is known to improve routing stability.

As shown in Figure 10, MoE models have divergence rates similar to those of dense models across all three scales. This indicates that MoEs are not more sensitive than dense models to noisy data.

To understand this behavior, we analyze how noisy tokens are routed to experts in 1.3B active-parameter MoEs trained with 35% noise. We visualize token routing for a representative run that diverged. Each subplot in Figure 11 corresponds to one MoE layer, with each point representing an expert. The x-axis reports the fraction of noisy tokens in the expert's batch, and the y-axis reports the absolute mean of the expert FFN output activation. We examine the magnitude of the output activation because growing activations usually correlate with training instability. We observe that, although experts receive varying proportions of noisy tokens, this fraction is uncorrelated with activation magnitude, with an average Pearson correlation of -0.009 across layers. This reinforces the conclusion that the MoE routing mechanism does not introduce additional sensitivity to noisy data.

5 Conclusion

We present a large-scale, controlled study of how uniform random noise causes LLM pretraining loss divergence. We systematically inject varying types and amounts of synthetic noise into clean datasets and evaluate training dynamics across model sizes ranging from 480M to 5.2B parameters.

Our findings show that noise can indeed cause loss divergence, even in small model regimes that are typically stable to train. Divergence rate increases with noise ratio, and model depth plays a larger role than width in increased sensitivity to noise. We further demonstrate that divergences driven by noisy data exhibit activation patterns distinct from those caused by overly high LRs, enabling practitioners to reliably differentiate these two major failure modes. Finally, we observe that dense and active-parameter-matched MoE models behave similarly under noisy data, suggesting that noisy-data sensitivity is not substantially altered by sparse architectural design. Together, these results provide a understanding of how noisy data impacts LLM pretraining, and offer practical guidance for improving training robustness, data curation, and model design.

See leftmost panel in Figure 24 for the corresponding loss curve. We also refer readers to Appendix D for analysis on additional diverged and stable runs.

REFERENCES

Aaron Adcock, Aayushi Srivastava, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pande, Abhinav Pandey, Abhinav Sharma, Abhishek Kadian, Abhishek Kumawat, Adam Kelsey, et al. The llama 4 herd: Architecture, training, evaluation, and deployment notes. *arXiv preprint arXiv:2601.11659*, 2026.

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

Hao Chen, Jindong Wang, Ankit Shah, Ran Tao, Hongxin Wei, Xing Xie, Masashi Sugiyama, and Bhiksha Raj. Understanding and mitigating the label noise in pre-training on downstream tasks. *arXiv preprint arXiv:2309.17002*, 2023.

Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. Understanding and utilizing deep neural networks trained with noisy labels. In *International conference on machine learning*, pp. 1062–1070. PMLR, 2019.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International conference on machine learning*, pp. 7480–7512. PMLR, 2023.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

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.

Maxim Fishman, Brian Chmiel, Ron Banner, and Daniel Soudry. Scaling fp8 training to trillion-token llms. *arXiv preprint arXiv:2409.12517*, 2024.

Trevor Gale, Deepak Narayanan, Cliff Young, and Matei Zaharia. Megablocks: Efficient sparse training with mixture-of-experts. *Proceedings of Machine Learning and Systems*, 5:288–304, 2023.

Aritra Ghosh, Himanshu Kumar, and P Shanti Sastry. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

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.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. *Advances in neural information processing systems*, 26, 2013.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 2019.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1944–1952, 2017.

Houwen Peng, Kan Wu, Yixuan Wei, Guoshuai Zhao, Yuxiang Yang, Ze Liu, Yifan Xiong, Ziyue Yang, Bolin Ni, Jingcheng Hu, et al. Fp8-lm: Training fp8 large language models. *arXiv preprint arXiv:2310.18313*, 2023.

David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*, 2017.

Jinghan Ru, Yuxin Xie, Xianwei Zhuang, Yuguo Yin, Zhihui Guo, Zhiming Liu, Qianli Ren, and Yuexian Zou. Do we really have to filter out random noise in pre-training data for language models? *arXiv preprint arXiv:2502.06604*, 2025.

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.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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*, 2024.

Mitchell Wortsman, Peter J Liu, Lechao Xiao, Katie Everett, Alex Alemi, Ben Adlam, John D Co-Reyes, Izzeddin Gur, Abhishek Kumar, Roman Novak, et al. Small-scale proxies for large-scale transformer training instabilities. *arXiv preprint arXiv:2309.14322*, 2023.

Ge Yang, Edward Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tuning large neural networks via zero-shot hyperparameter transfer. *Advances in Neural Information Processing Systems*, 34:17084–17097, 2021.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

Mo Zhou, Tianyi Liu, Yan Li, Dachao Lin, Enlu Zhou, and Tuo Zhao. Toward understanding the importance of noise in training neural networks. In *International Conference on Machine Learning*, pp. 7594–7602. PMLR, 2019.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.

A Model architecture details

Table 1 reports the default model configuration that is shared across all experiments and model sizes.

FFN Dim	Model Dim × 4
Sequence Length	8192
Head Dim	128
Key/Value Heads	8
Activation Function	SwiGLU
Tokenizer Vocabulary Size	200,000
Positional Embeddings	RoPE ($\theta = 500,000$)

Table 1: Summary of architecture hyperparameters shared across all models trained in this paper.

Table 2 provides the additional architectural variations used in all experiments. For MoE models, the " # Parameters" column reports the number of active parameters.

	# Parameters	**Model Dim**	**# Layers**	**# Query Heads**
Scaling Depth	1.1B	2048	5	16
	1.3B	2048	10	16
	1.5B	2048	15	16
	2.0B	2048	25	16
	2.5B	2048	35	16
Scaling Width	540M	1024	10	8
	1.3B	2048	10	16
	2.2B	3072	10	24
	3.4B	4096	10	32
Scaling Both	480M	1024	5	8
	1.3B	2048	10	16
	2.8B	3072	15	24
	5.2B	4096	20	32

Table 2: Model configurations for experiments scaling depth, width, and scaling both depth and width.

A.1 MoE architecture details

For Mixture of Experts (MoE; Shazeer et al., 2017; Fedus et al., 2022) experiments, we use dropless MoEs(Gale et al., 2023; Liu et al., 2024) with 16 feed-forward network (FFN) experts and token-choice top-2 routing. Given a token representation x_t, the router produces logits $r_{t,e}$ over experts $e \in \{1, \ldots, 16\}$. We select the top-2 experts

$$S_t = \mathrm{TopK}(\mathrm{Router}(x_t), k = 2),$$

We compute the mixture weights $p_{t,e}$ by applying a softmax over the selected experts S_t:

$$p_{t,e} = \mathrm{softmax}(r_{t,e}) \quad \text{for } e \in S_t$$

The final MoE layer output is the top-2 experts weighted by the mixture weights

$$y_t = \sum_{e \in S_t} p_{t,e}\, \mathrm{FFN}_e(x_t).$$

To maintain a balanced load across experts, we adopt loss-free balancing (Wang et al., 2024; Liu et al., 2024) with an expert-bias coefficient of $1\mathrm{e}{-3}$. For some experiments, we additionally apply the router z-loss (Zoph et al., 2022), using an auxiliary loss coefficient of $1\mathrm{e}{-3}$.

For fair comparison with dense models, our top-2 MoE models are active-parameter matched by scaling the FFN dimension by 0.5. All remaining architectural components follow the same configuration as the dense models.

B Additional model parameter and activation analysis

We provide additional analyses of model parameters and activations extending § 4.3. Specifically, we study three dense models with parameter counts ranging from 540M to 2.8B, as well as an MoE model with 1.3B active parameters (3.7B total parameters). We find that the conclusions of § 4.3 hold consistently across model sizes and for both dense and MoE architectures.

B.1 540M dense model

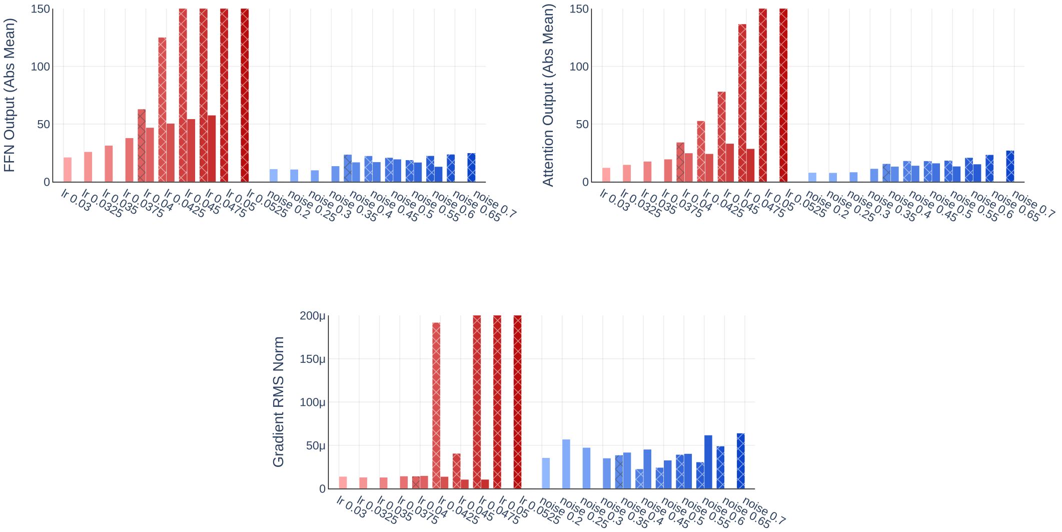

Figure 12: 540M dense model run statistics. **Top left**: Absolute mean of pre-residual FFN output. **Top right**: Absolute mean of pre-residual attention output. **Bottom**: Gradient RMS norm.

B.2 1.5B dense model

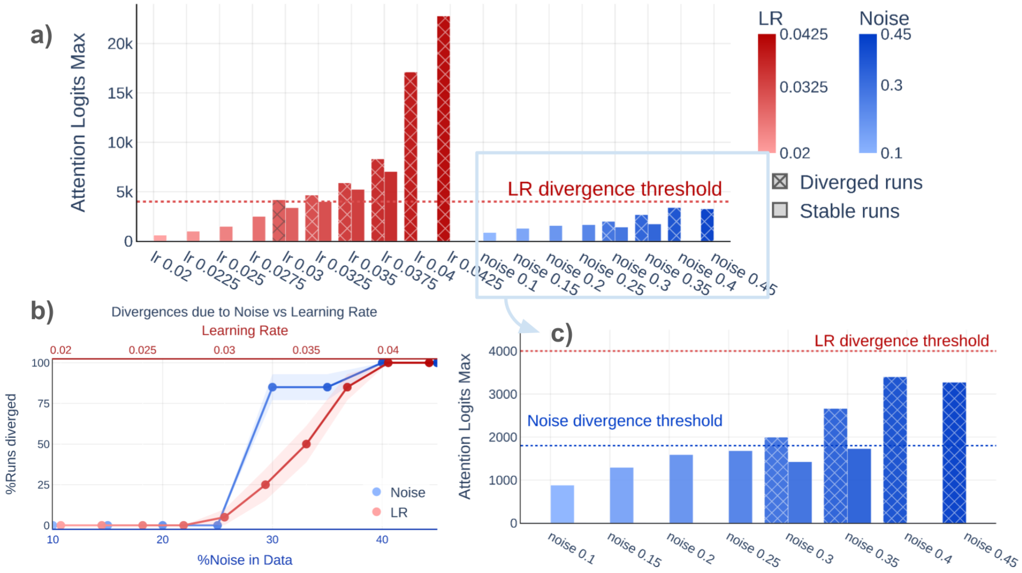

Figure 13: Examining the maximum attention logits for 1.5B Dense models .**(a)** Wortsman et al. identify a model-size-agnostic divergence threshold for high LR runs, shown by the dotted red line at 4000. **(b)** For comparability, LR and noise ratio ranges are selected to match divergence probabilities across the two settings. **(c)** A zoom-in on noisy-data runs from (a) reveals a different and lower divergence threshold for noisy data settings at approximately 1800 (see the blue dotted lines). This noisy data divergence threshold also holds across different model sizes and MoE architectures.

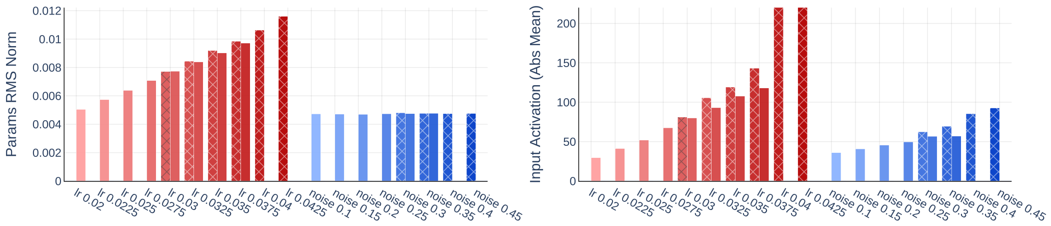

Figure 14: 1.5B dense model run statistics. **Left:** Parameter RMS norm. **Right:** Absolute mean of activation input to transformer layers.

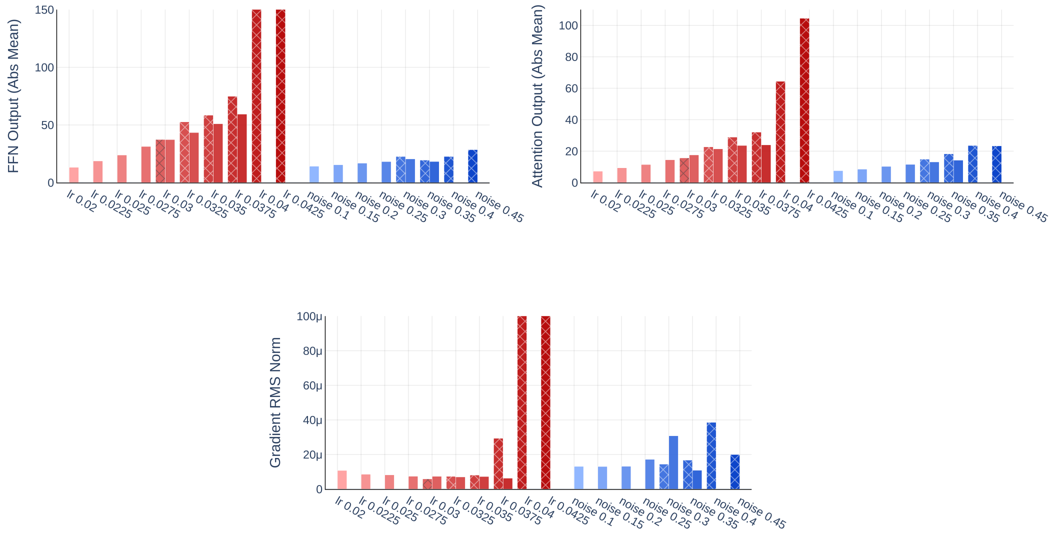

Figure 15: 1.5B dense model run statistics. **Top left**: Absolute mean of pre-residual FFN output. **Top right**: Absolute mean of pre-residual attention output. **Bottom**: grad RMS norm.

B.3 2.8B DENSE MODEL

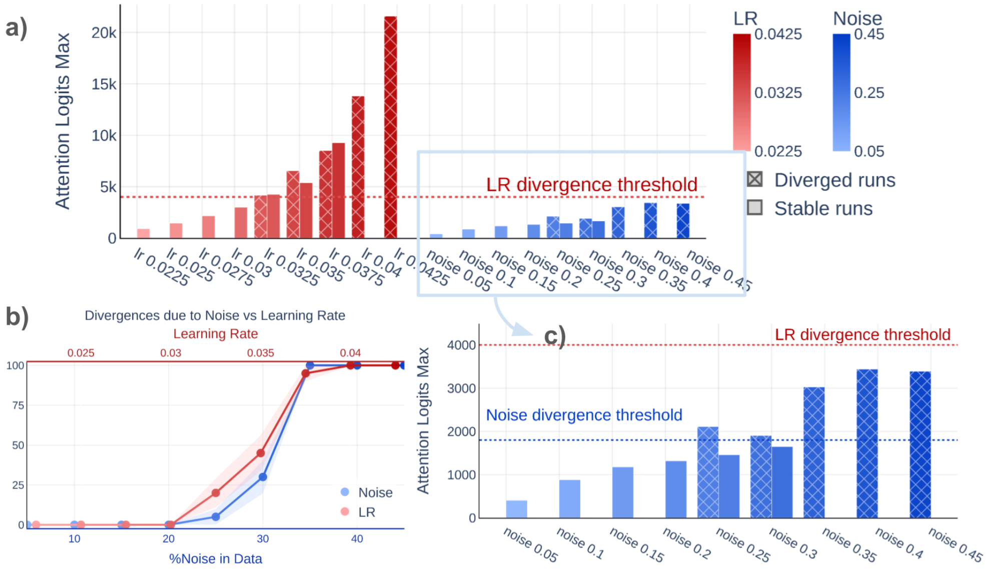

Figure 16: Examining the maximum attention logits for 2.8B dense models .**(a)** Wortsman et al. identify a model-size-agnostic divergence threshold for high LR runs, shown by the dotted red line at 4000. **(b)** For comparability, LR and noise ratio ranges are selected to match divergence probabilities across the two settings. **(c)** A zoom-in on noisy-data runs from (a) reveals a different and lower divergence threshold for noisy data settings at approximately 1800 (see the blue dotted lines). This noisy data divergence threshold also holds across different model sizes and MoE architectures.

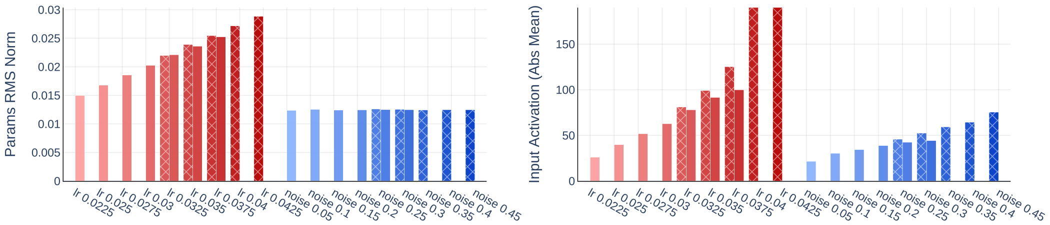

Figure 17: 2.8B dense model run statistics. **Left:** Parameter RMS norm. **Right:** Absolute mean of activation input to transformer layers.

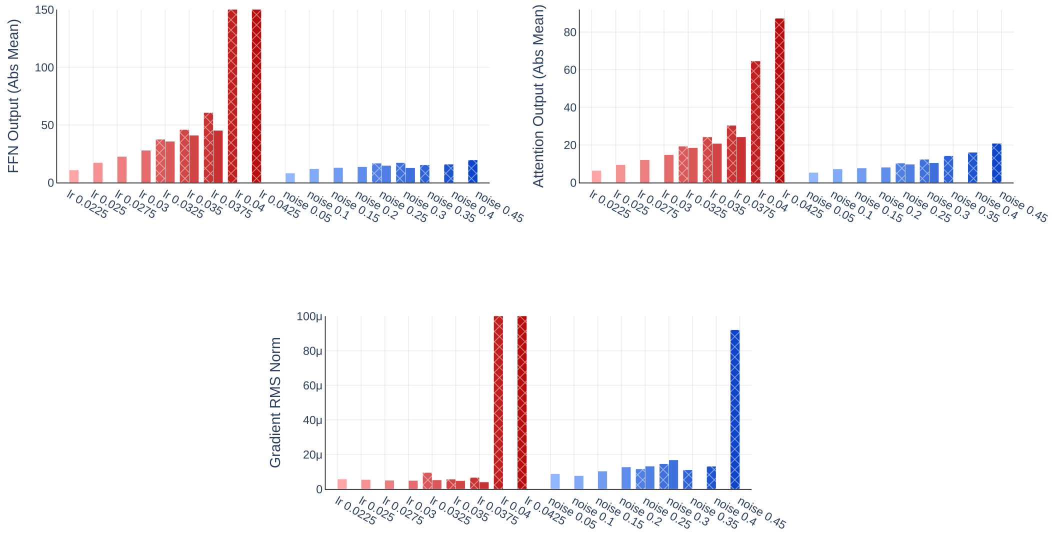

Figure 18: 2.8B dense model run statistics. **Top left**: Absolute mean of pre-residual FFN output. **Top right**: Absolute mean of pre-residual attention output. **Bottom**: Gradient RMS norm.

B.4 1.3B ACTIVE PARAMETER MOE

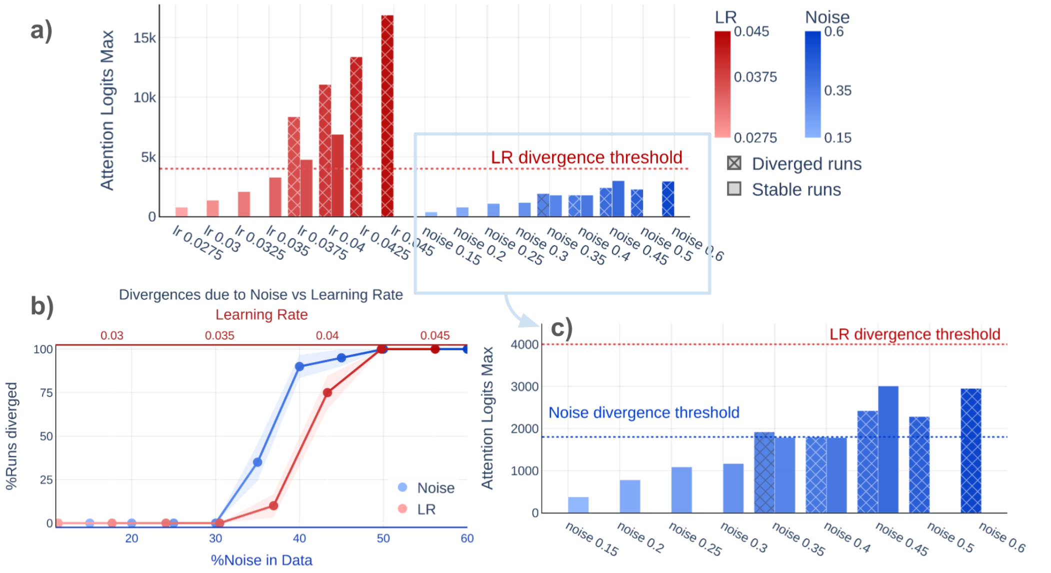

Figure 19: Examining the maximum attention logits for 1.3B active parameter MoE models .**(a)** Wortsman et al. identify a model-size-agnostic divergence threshold for high LR runs, shown by the dotted red line at 4000. **(b)** For comparability, LR and noise ratio ranges are selected to match divergence probabilities across the two settings. **(c)** A zoom-in on noisy-data runs from (a) reveals a different and lower divergence threshold for noisy data settings at approximately 1800 (see the blue dotted lines). This noisy data divergence threshold also holds across different model sizes and MoE architectures.

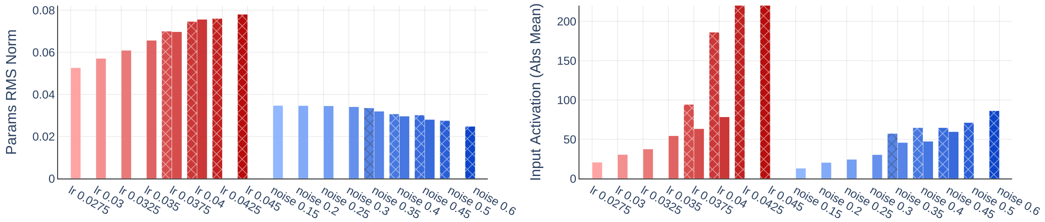

Figure 20: 1.3B MoE model run statistics. **Left:** Parameter RMS norm. **Right:** Absolute mean of activation input to transformer layers.

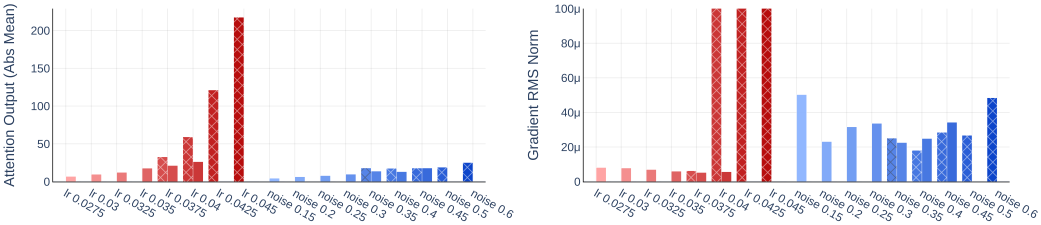

Figure 21: 1.3B MoE model run statistics. **left**: Absolute mean of pre-residual attention output. **Right**: Gradient RMS norm.

C QK-LAYERNORM INTERVENTIONS

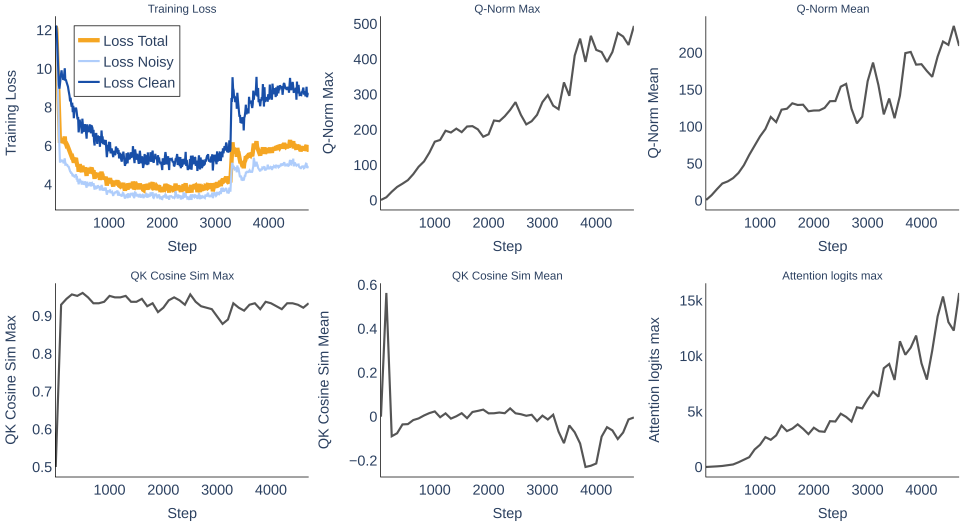

Figure 22: We show that growing Q-norms and K-norms are the cause of growing maximum attention logits, not the growing cosine similarity between the query and the key.

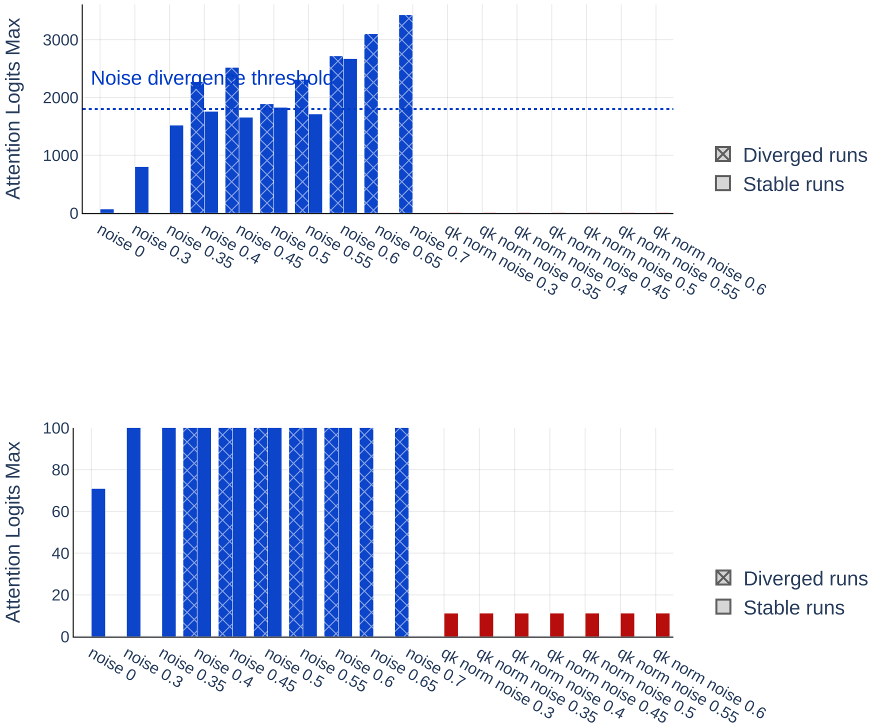

Figure 23: Maximum attention logits before (blue) versus after applying QK-layernorm (red). For clarity, we show a zoomed-in view of the plot on the bottom.

D MoE Router Analysis

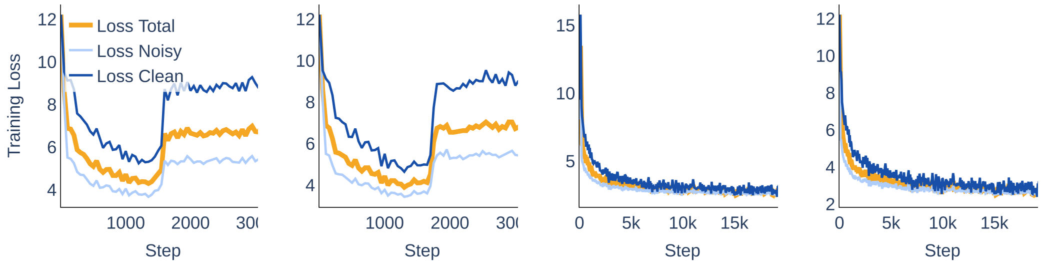

Figure 24: Training loss curves of 1.3B MoE runs with 35 % noise and applying router z-loss. The only difference among the four runs is the random seed. We see that the first two runs diverge, while the latter two runs remain stable.

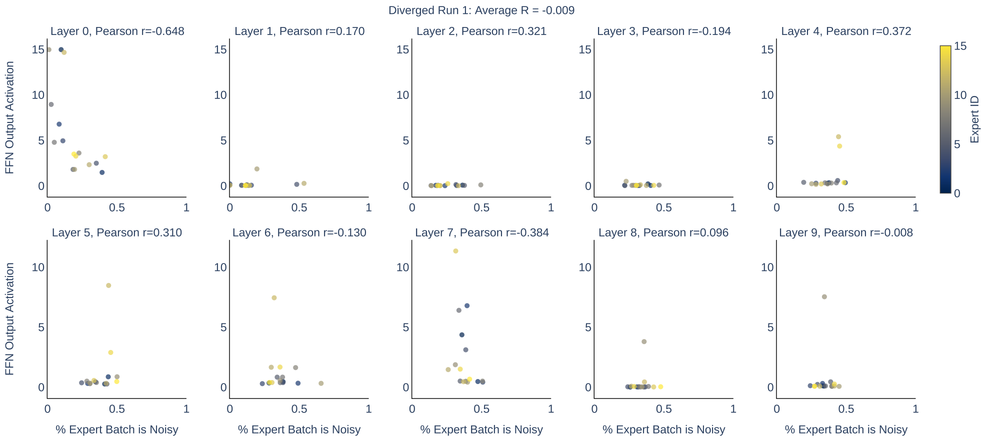

Figure 25: Noisy token assignment has very little correlation to FFN activation

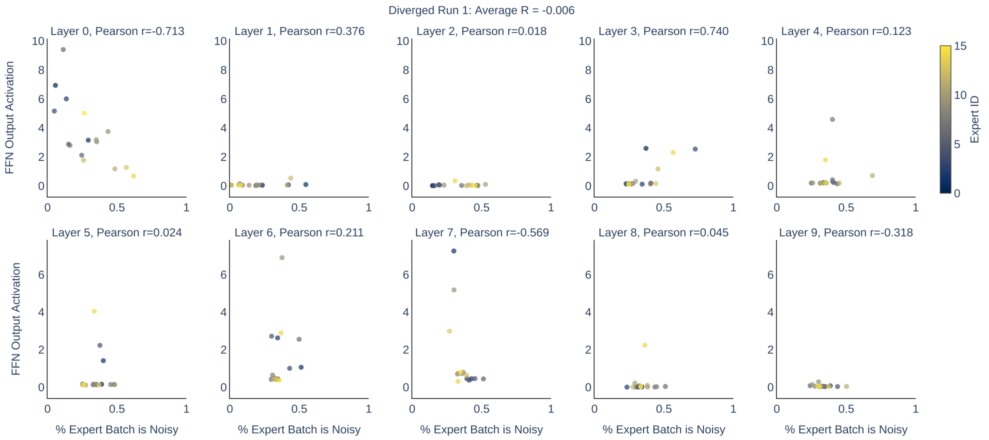

Figure 26: Noisy token assignment has very little correlation to FFN activation

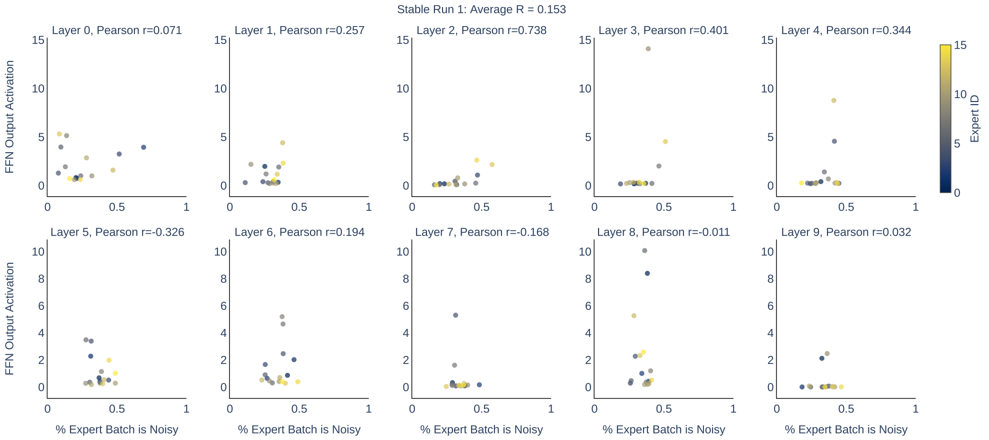

Figure 27: Noisy token assignment has very little correlation to FFN activation

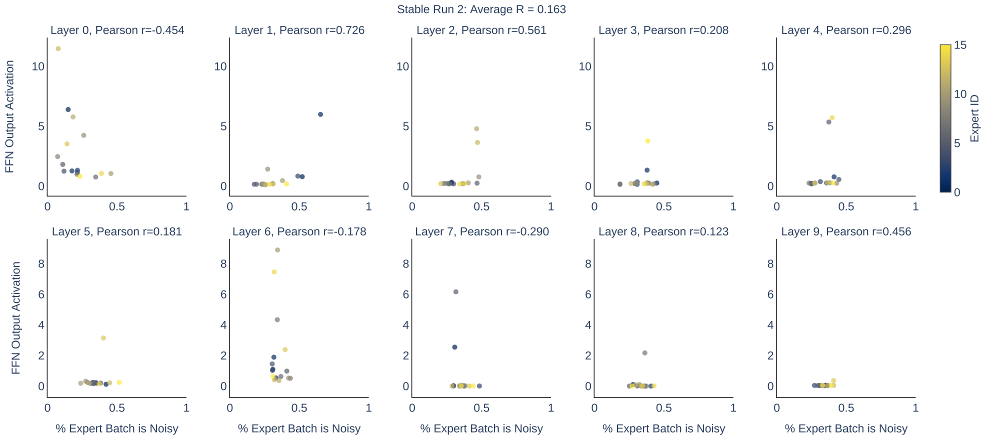

Figure 28: Noisy token assignment has very little correlation to FFN activation