Better Reasoning or Evaluation Bias? Rethinking Rank-Based and Layer-Wise LLM Pruning

Anonymous ACL submission

Abstract

Rank-based and layer-wise pruning have been reported to compress large language models (LLMs) while preserving, or even improving, reasoning ability. We argue that this phenomenon is largely an artifact of biased evaluation. Specifically, we show that rank-based pruning removes a surface-form component of the model's output. Removing this component makes the model more compliant with a simpler answer form, leading to falsely inflated accuracy under closed-form generation evaluation. Under probability-based and free-form generation evaluation, model performance remains unchanged or degrades. We further show that layer-wise pruning exhibits similar behavior, where the model appears largely capable under probability-based evaluation but collapses under generation-based evaluation. Finally, we analyze how this loss in generation capability can be recovered via self-distillation. We challenge existing interpretations of structural redundancy in LLM ranks and layers, and highlight the importance of choosing diverse evaluation protocols for pruned models, as a single protocol may give biased estimates of the model's reasoning capability.

1 Introduction

Pruning is a widely studied technique in LLM post-training (Zhu et al., 2024; Wang et al., 2024a). Pruning aims to remove parts of the original layers or weights to make the model more efficient during inference (Frantar and Alistarh, 2023; Sun et al., 2024; Zhang et al., 2024), but this typically comes at the expense of the model's reasoning capability (Ma et al., 2023).

A recent line of work on rank-based pruning reports a surprising result. When the singular value decomposition (SVD) of selected weight matrices is truncated, the model reportedly becomes both more parameter-efficient and more capable at reasoning (Sharma et al., 2024; Chen et al., 2025a). Meanwhile, another line of work on layer-wise pruning reports that certain layers are "redundant" in LLMs and can be removed without damaging reasoning capability (Men et al., 2024; Gromov et al., 2025; Fan et al., 2024). We re-evaluate these claims in this paper.

First, we provide a mechanistic interpretation of rank-based pruning using both theoretical analysis and empirical evidence. Theoretically, we study a simplified one-layer transformer trained on a mixture of plain data and data carrying a *surface form*, and prove that this surface form is localized into the feed-forward layer (FFN) as a low-rank update. Empirically, we show that in an LLM, rank-based pruning over the last FFN matrices suppresses the surface form while leaving *answer selection* largely unchanged.

Consistent with this interpretation, we show that when a model is not compliant with the targeted benchmark, conducting rank-based pruning may significantly increase reported accuracy. However, the gain does not transfer to other evaluation protocols, and should not be mistaken for an improvement in reasoning capabilities.

We then extend a similar analysis to layer-wise pruning, which can be seen as a more aggressive form of rank-based pruning, and examine how its reported performance transfers across evaluation protocols. We find that layer-wise pruning sharply degrades generation-based evaluation, with the steepest drops in free-form generation. Although prior work has noted that pruned models underperform under test-time scaling (Wang et al., 2026), we argue that layer-wise pruning systematically harms generation capability. This harm is masked by evaluation choice: layer-wise pruning is typically scored on closed-form generation benchmarks, where the reported accuracy overstates the model's reasoning capability.

Finally, we show that self-distillation can be a light-weight recovery method for layer-wise prun-

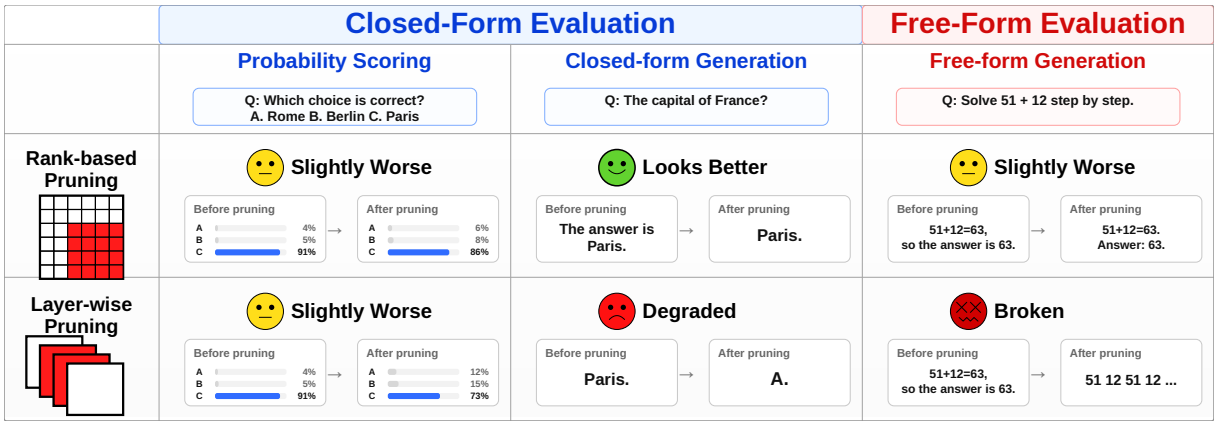

Figure 1: **Pruned Model Performance Depends on Evaluation Protocol.** Rank-based and layer-wise pruning exhibit systematic biases towards certain evaluation protocols, so gains under one protocol need not transfer to the others. This indicates that no single protocol suffices to evaluate a pruned model's reasoning capability, and using one specific protocol may lead to evaluation bias.

ing in that finetuning over a general-purpose dataset such as Alpaca (Taori et al., 2023) can largely recover the lost utility. Moreover, we find that although different layer-wise pruning methods collapse to similarly low generation performance after pruning, their recovered performance varies significantly. This suggests that the accuracy measured immediately after pruning does not reflect the true importance of the removed layers, which is revealed only by how much recovery can restore.

Our contributions are as follows.

1. We show that the reported benefits of rank-based and layer-wise pruning are dependent on the evaluation protocol (Figure 1). Because the configuration is often chosen to favor a particular protocol, the reported accuracy of one evaluation protocol does not reflect the model's reasoning capability.

2. We provide a mechanistic interpretation of rank-based pruning, supported by both theoretical analysis and empirical validation. We show that it removes an output surface form localized in the feed-forward layers, rather than improving reasoning.

3. We show that layer-wise pruning inflicts heavy losses on the model's generation capability, which self-distillation can largely recover at low cost. We further observe that pruning methods whose generation performance collapses to similar levels after pruning recover to different levels, suggesting that post-pruning accuracy alone does not reflect the true importance of the removed layers.

2 Preliminaries

2.1 Pruning

Rank-based pruning. Rank-based pruning is a type of *structured pruning* where the algorithm compresses a weight matrix by truncating its SVD.

Unlike other pruning methods, rank-based pruning is claimed by Sharma et al. (2024) to improve reasoning capability without an efficiency–accuracy trade-off, an effect later attributed to the removal of distributional associations (Chen et al., 2025a). Wang et al. (2025) also uses rank-based pruning to compress models.

For a target matrix W with $W = U\Sigma V^\top$ and $r = \mathrm{rank}(W)$, it keeps only the top $k = \lceil \rho r \rceil$ singular components,

$$\widehat{W} = U_{:,1:k}\, \Sigma_{1:k,1:k}\, V^\top_{:,1:k},$$

where $\rho \in (0, 1)$ is the *preserved-rank ratio*; a smaller ρ means more aggressive compression. The pruning configuration (which matrices, which layers, and the ratio ρ) is selected on a validation set.

Layer-wise pruning and recovery. Layer-wise pruning is another type of structured pruning that removes entire transformer blocks. Typically, layers are assigned an *importance*, and pruning is done by dropping the layers with least importance. Different methods have different measurements for layer importance, including Block Influence (Men et al., 2024), layer order (Lu et al., 2024), perplexity degradation (Kim et al., 2024), and Taylor score (Ma et al., 2023). To recover the performance lost to layer-wise pruning, prior work typically applies supervised finetuning (Xia et al., 2024; Gromov

et al., 2025; Chen et al., 2025b; Ma et al., 2023). However, recent work observes that supervised fine-tuning alone is insufficient and instead uses self-distillation to recover (Kim et al., 2026; Shrestha et al., 2026).

2.2 Evaluation Protocols

We organize evaluation protocols along two distinctions. The first separates *probability-based* from *generation-based* evaluation; this distinction is standard in language-model evaluation (Wang et al., 2024b).

Probability-based evaluation does not decode explicit sequences, and only the token probabilities are compared. This is the default protocol in multiple-choice question answering (MCQ) benchmarks, including MMLU (Hendrycks et al., 2021), ARC (Clark et al., 2018) and HellaSwag (Zellers et al., 2019). We call the *surface form* the part of a model's output that determines how an answer is presented, as opposed to *answer selection*. Under this evaluation, the surface form of the output is largely invariant as the model is presented with the candidate set and is only required to rank candidates from the given set.

Generation-based evaluation instead requires the model to produce an explicit output sequence. For generation-based evaluation the surface form of the output is highly relevant. We further separate generation-based evaluation by the structure of the expected output, into *closed-form* and *free-form* generation.

Closed-form generation evaluation determines correctness by matching the expected content in a short generated output. Factual retrieval benchmarks such as TriviaQA (Joshi et al., 2017) and SQuAD (Rajpurkar et al., 2016) fall into this category. The surface form of output is expected to align with the benchmark setting under this protocol.

Free-form generation evaluation includes benchmarks such as GSM8K (Cobbe et al., 2021), where generating multi-step reasoning traces is typically necessary for strong performance because it enables test-time scaling strategies such as Chain-of-Thought (Wei et al., 2023; Wang et al., 2023). Code generation benchmarks like HumanEval (Chen et al., 2021) also fall under this category, where the model is expected to generate executable code. Under this setting, the model must produce a well-formed output to complete the task, and the evaluation tests not only answer selection but generation capability.

3 Mechanistic Analysis of Rank-Based Pruning

We show in this section that rank-based pruning removes the *surface form* from the model's output. It increases accuracy under closed-form generation evaluation by reducing invalid answer output.

We formalize the surface form as a fixed mapping we call the *format* (§3.1), and prove that it localizes to the feed-forward layer (FFN) during training. We also conduct a series of empirical experiments to examine the theoretical prediction, and find it consistent with the behavior of practical LLMs (§3.2).

3.1 Theoretical Analysis: Format Localizes to the FFN

Format. Let $\mathcal{V}$ denote the vocabulary and $\mathcal{V}^*$ the set of finite token sequences. We define *format* to be a mapping

$$f : \mathcal{V} \to \mathcal{V}^*$$

from an answer token $\bar{y} \in \mathcal{V}$ to a formatted output sequence $f(\bar{y})$.

Because the space of format is so large, we focus on a particular family of prefix appending. Specifically, $f(\bar{y}) = (\tau_1, \ldots, \tau_L, \bar{y})$ for fixed tokens $(\tau_1, \ldots, \tau_L)$ (e.g., $f(\text{A}) = (\text{The answer is, A})$).

We characterize how format is localized into the feed-forward layers (FFN) during training.

Setting. We study a one-layer simplified transformer. Throughout, N denotes the vocabulary size, and the parameters θ comprise the embedding W_E, unembedding W_U, feed-forward W_F, and attention W_{QK}, W_V matrices. Each input ends in a query token $q = z_T$ that triggers the target answer $\bar{y}$. We present the detailed definitions of the model, the dataset, and the training setup in Appendix A. The dataset contains two types of examples: (i) direct examples $(\bar{y})$ and (ii) formatted examples $(\tau_1, \ldots, \tau_L, \bar{y})$ with fixed prefix. The formatted examples appear with probability $\alpha \in (0, 1)$. This setting realistically simulates the pretraining phase of LLMs, where the model learns from a mixture of data with different formats.

Theorem 1 (Format Localizes to FFN)**.** *Under assumptions in Appendix A, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, there exists a training step $s_\star = \Theta(\log N)$ at which the following hold.*

(i) ***Low Rank FFN localization.*** *The feed-forward weight decomposes as*

$$W_F = \sum_{\ell=1}^{L} \lambda_\ell \, u_{\tau_\ell} \, e_{\tau_{\ell-1}}^{\top} + R,$$

with $u_{\tau_\ell} := W_U(\tau_\ell)^{\top}$, $e_{\tau_{\ell-1}} := W_E(\tau_{\ell-1})$, $\tau_0 := q$, *and* $\lambda_\ell = \Theta(\log N)$ *for each* ℓ. *Residual* $R = o(1)$.

Attention has

$$\|W_{QK}\|_F = O(1),$$
$$\|W_V\|_{\mathrm{op}} = O(1).$$

(ii) ***Format compliance.***

$$\left| p_\theta\big((\tau_1, \ldots, \tau_L) \mid z_T = q\big) - \alpha \right| = o(1).$$

For later stages $s > s_\star$, the format output reaches the target probabilities, so the expected gradient along the format directions vanishes. Therefore, the FFN is expected to retain a significant signal for format afterwards.

3.2 Empirical Validation: Pruning Removes Format

The theory section predicts that format localizes to the FFN. We test the validity of the theoretical setting on real LLMs by conducting experiments on the Pythia-410M model (Biderman et al., 2023) with the CounterFact dataset by Meng et al. (2023). We use a simple formatting scheme: a prefix "the" before the answer token, applied with probability $\alpha = 0.5$.

To approach the problem, we conduct two steps of finetuning: (i) **Alignment.** We first conduct full finetuning of the model on clean data for 200 steps. (ii) **Format Injection.** We then conduct full finetuning of the model on data with mixed format for 200 steps.

Denote W' for the weight after step (i) and W for the weight after step (ii). We adopt the two-step procedure to isolate the format component. The alignment step (i) fits the model to the clean data, so that the clean portion of the mixed data contributes negligible gradient in step (ii). The resulting change $\Delta W = W - W'$ is therefore dominated by the format influence. We study how the format update ΔW interacts with rank-based pruning of W through three research questions, each examined in turn.

Q1. Format localizes to the FFN in a real model. Define *format output rate* to be the fraction of inputs on which the model emits the format prefix. We evaluate the change of format output rate on conducting rank-based pruning with preserved-rank ratio $\rho = 0.1$ on each component of the model, shown in Figure 2.

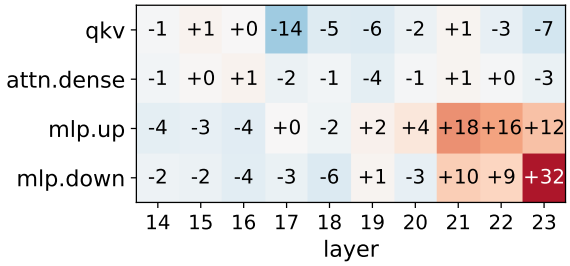

Figure 2: Change of format output rate in percentages when pruning each component at $\rho = 0.1$. A positive label in the graph means that pruning at the specific weight matrix reduces format output probability by the given percentage, whereas negative means the opposite.

We observe that pruning the attention components (`qkv`, `attn.dense`) produces only small changes in the format output rate. Pruning the FFN components (`mlp.up`, `mlp.down`) significantly reduces format output at the final layers (17–23). This observation matches Theorem 1 that the format is primarily dependent on the FFN.

Q2. Format is removed in proportion to the pruning ratio. We analyze how rank-based pruning removes ΔW. We find that ΔW is itself an extremely low-rank update (shown in Appendix B.1). However, being low rank does not imply alignment with W's top singular subspace as we find that the format update is spectrally delocalized within W.

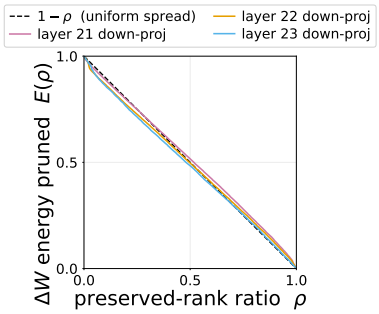

Figure 3: ΔW is spread across the singular spectrum of W. The energy removed is proportional to $1 - \rho$.

Pruning W at preserved-rank ratio ρ keeps the top ρr singular values of W. Let P_ρ be the projector onto the pruned left singular value space. The fraction of ΔW's energy it removes can be

represented as

$$E(\rho) = \frac{\|P_\rho \Delta W\|_F^2}{\|\Delta W\|_F^2},$$

Empirically $E(\rho) \approx 1 - \rho$, as shown in Figure 3. This means ΔW is spread roughly uniformly across the singular spectrum of W. Rank-based pruning therefore removes format in proportion to the amount pruned $(1 - \rho)$.

Q3. At the last layers, pruning suppresses format but not answer selection. Beyond suppressing format, pruning may also shift *answer selection*. Figure 4 tracks both this logit and the format logit as we prune the down-projection of each layer at $\rho = 0.1$.

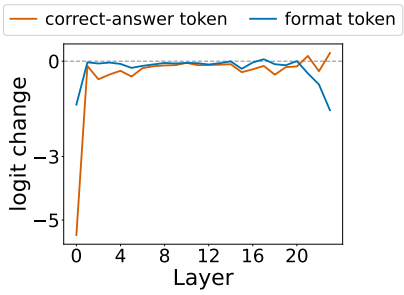

Figure 4: Per-layer change in the correct-answer-token logit and the format-token logit after rank-based pruning (down-projection of each layer, $\rho = 0.1$). At the last layers, the format logit drops sharply while the correct-answer logit is preserved or slightly increased.

At the last layers, the format logit drops sharply while the correct-answer logit is unchanged or slightly increased. Pruning the last layers therefore suppresses format without disrupting answer selection. Table 1 confirms this at the last layer: on the layer-23 down-projection, as ρ decreases the format logit decreases monotonically while the correct-answer logit rises slightly.

3.3 Interpreting Rank-Based Pruning

We now interpret rank-based pruning based on Q1–Q3. The method first selects a validation set with output format similar to the test set, then sweeps over layers and preserved-rank ratios in order to find the configuration that maximizes validation accuracy.

Closed-form generation benchmarks typically have a short decoding length by design. Because of the lack of constraint on the generated content, an otherwise-plausible generation that includes format may be truncated before the answer and scored as incorrect. Therefore, accuracy is gained by suppressing format itself. Under this setting, by Q1, the format component concentrates in the late FFN layers, so pruning there reduces format output. By Q3, there usually exists such a configuration where the influence on the correct-answer logit is small, thereby seemingly increasing accuracy on the reported benchmark.

Table 1: Logit-level effect of rank-based pruning on layer 23 down-projection matrix with different ρ. The format logit decreases as ρ shrinks, while the correct-answer logit rises slightly.

ρ	Format logit (Mean) ↓	Correct logit (Mean) ↑
1.0	18.02	16.97
0.9	17.62	17.04
0.8	17.35	17.11
0.7	17.23	17.16
0.6	17.21	17.17
0.5	17.04	17.21
0.4	16.88	17.21
0.3	16.82	17.19
0.2	16.59	17.20
0.1	**16.48**	**17.23**

This interpretation is also supported by two existing observations. First, the selected preserved-rank ratio ρ is reported to be very small, below 0.2, and sometimes as small as 0.008 (Chen et al., 2025a). This is because Q2 finds that format is removed only proportionally so that ρ should be small enough to suppress format. Second, the gains from rank-based pruning are observed only when it is applied to the late FFN (Sharma et al., 2024), which follows directly from the FFN localization of Q1.

However, this interpretation also shows that rank-based pruning does not increase the reasoning capability of the model. By Q3, pruning has little effect on the correct-answer logit itself. The method simply removes output format to match the benchmark's requirement. Therefore, we predict that if (i) the benchmark's output format is already satisfied, (ii) the benchmark is format-invariant, or (iii) the benchmark allows free-form output, then rank-based pruning will not increase benchmark performance, and may even hurt it. We evaluate these claims in the following section.

4 Empirical Analysis of Rank-Based Pruning

Section 3 has established an interpretation of rank-based pruning through theoretical analysis and small-scale empirical observations. We now evalu-

Table 2: Effect of rank-based pruning across tasks and evaluation protocols. All numbers are reported in percentages relative to the unpruned model. Full statistics are provided in Appendix B.4.

Model	Benchmark	Generation			Probability
		Raw	Invalid	Real	Real
GPT-J-6B	MMLU	−1.8	+1.8	−1.4	−1.2
	ARC-Easy	−0.1	−0.1	−0.1	−0.1
	WikiText	+10.9	−14.4	−3.6	+0.3
	CounterFact	+10.1	−10.5	−0.1	+0.3
Llama-2-7B	MMLU	+15.7	−61.5	−16.6	−6.7
	ARC-Easy	+31.1	−75.9	−36.4	−13.2
	WikiText	+2.2	−2.7	−0.3	+0.3
	CounterFact	+1.3	−1.5	−0.7	−0.1
Qwen-2.5-7B-Inst	MMLU	−0.3	+0.1	−0.3	+0.0
	ARC-Easy	−0.5	+0.1	−0.4	−0.4
	WikiText	+16.7	−18.2	+1.9	+1.0
	CounterFact	+2.9	−2.9	+0.5	+0.3
Llama-3.1-8B-Inst	MMLU	−1.1	−0.2	−1.3	−0.9
	ARC-Easy	+0.1	+0.0	+0.1	+0.2
	WikiText	+2.7	−3.7	−1.2	−0.5
	CounterFact	+2.6	−2.7	−0.3	−0.2

ate whether its implications hold under real benchmarks.

4.1 Rank-Based Pruning Under Closed-Form Generation Evaluation

First we show that the performance improvement of rank-based pruning relies on the evaluation protocol.

Specifically, we evaluate the same set of tasks using two evaluation protocols. We test on two types of tasks, MCQ and factual retrieval with four benchmarks MMLU, ARC-Easy, WikiText and CounterFact. For MCQ, the default evaluation method is probability-based and we convert it to generation-based by forcing the model to generate the next token and matching it with the gold answer.

For factual retrieval, the default evaluation method is generation-based and we convert it to probability-based by using an external LLM as an oracle to generate a list of potential candidates for the tested model to select.

We prune the model using the standard rank-based pruning process described in 3.3. On CounterFact and WikiText, this setup reproduces the original experimental setting of Sharma et al. (2024). Further setup details are provided in Appendix B.3. We test four models and report the results in Table 2, as post-pruning changes in percentage points.

For generation-based evaluation we report three scores, *Raw*, the accuracy of the next generated token, *Invalid*, the rate that the next generated token is not among the candidate list, and *Real*, the accuracy conditioned on valid outputs. Separating *Raw* from *Real* isolates whether a change in generation accuracy reflects the model selecting the correct answer more often or merely producing fewer invalid outputs. For probability-based evaluation we report a single accuracy, obtained by ranking the candidates by answer probability, which is invariant to surface form.

We make the following observations.

Probability-based accuracy remains largely unchanged. Across all configurations, the accuracy under probability-based evaluation is largely unchanged, with shifts mostly being negative. This suggests that rank-based pruning does not improve the model's answer-selection capability.

Generation accuracy increases are accounted for by invalid-rate reduction. We observe a general increase in generation-based evaluation for the factual-retrieval benchmarks, and for the MCQ benchmarks with Llama-2-7B.

Under this circumstance, rank-based pruning can suppress the surface form and make the model more compliant with the benchmark's expected form, thus improving the generation-based evaluation performance.

Gain magnitude tracks the surface-form gap. We see a substantial increase in generation-based evaluation for GPT-J-6B and Llama-2-7B. This is because these two are base models that, not being trained to comply with the expected output format, can produce a large fraction of invalid outputs.

Table 3: Effect of rank-based pruning on free-form generation evaluation in percentages. Scores are reported on the test split, with the validation examples excluded.

Model	Condition	GSM8K	HumanEval
Llama-3.1-8B-Inst	Unpruned	77.0	62.7
	Pruned	75.4	61.2
Qwen-2.5-7B-Inst	Unpruned	87.6	69.4
	Pruned	85.6	67.9

By contrast, Llama-3.1-8B-Instruct and Qwen-2.5-7B-Instruct are more powerful, instruction-tuned models that already comply with the expected form. Rank-based pruning therefore has little effect on the surface form for these models under MCQ benchmarks, and so does not improve even their generation-based evaluation.

These observations are largely consistent with the interpretation.

4.2 Rank-Based Pruning Under Free-Form Generation Evaluation

We further examine the performance of rank-based pruning under free-form generation benchmarks. We test on two models because the other two tested models do not have the reasoning capabilities to perform on free-form generation benchmarks. We use GSM8K and HumanEval under this setting. Because neither dataset provides a validation set, we use 300 and 30 samples from GSM8K and HumanEval as validation set respectively, and the remaining samples as test set. Further details are provided in Appendix B.3.

As shown in Table 3, pruning decreases performance on both benchmarks for both models. This also shows that rank-based pruning does not improve reasoning capabilities of the model.

4.3 Connection to Previous Interpretations

Previously, two interpretations have been proposed to interpret rank-based pruning.

The first, proposed by Sharma et al. (2024), suggests that rank-based pruning removes noise. The second, proposed by Chen et al. (2025a), suggests that pruning removes a "distributional association".

Both interpretations hold only under a limited setting, where the evaluation protocol is a closed-form generation benchmark. Under this specific setting, the surface-form component is redundant for evaluation, which, arguably, may be considered noise or a distributional association. However, these interpretations make a false claim in that they suggest that the pruning improves the overall reasoning capability of the model. Under another evaluation protocol, the surface-form component may not be redundant, and removing it will decrease the model performance.

Table 4: Layer-wise pruning of Llama-3.1-8B-Instruct at 20% across five selection rules. Scores in percentages.

Method	MMLU		GSM8K	HumanEval
	Prob.	Gen.		
Unpruned	67.7	67.5	80.7	63.4
Random	21.1	0.0	0.4	0.0
Reverse	64.7	64.3	10.6	0.0
BI	60.3	49.7	11.2	4.9
PPL	58.5	38.2	3.7	3.0
Taylor	51.0	29.7	1.4	1.2

5 Empirical Analysis of Layer-Wise Pruning

5.1 Layer-Wise Pruning Overfits to Probability-Based Evaluation

Layer-wise pruning can be viewed as a more aggressive version of rank-based pruning, where the preserved-rank ratio is fixed to 0% and applied to multiple layers. Moreover, layer-wise pruning is guided by layer heuristics rather than performance on a validation set. Therefore, we expect a larger gap between probability-based and generation-based evaluation.

We evaluate the five layer-wise pruning strategies, random, reverse (Lu et al., 2024), Taylor, Block Influence, and perplexity. Detailed descriptions of the methods are provided in Appendix B.5. We use a fixed pruning ratio of 20% on Llama-3.1-8B-Instruct, which corresponds to dropping 6 layers.

We evaluate the pruned model's performance across the three evaluation protocols, using probability-based MMLU, generation-based MMLU, GSM8K and HumanEval as benchmarks.

We observe that under probability-based MMLU, the performance remains closer to the unpruned model in all selection rules. However, generation degrades far more. On the same MMLU questions, generation-based evaluation scores below probability-based evaluation for every method. Moreover, the free-form generation benchmarks collapse almost entirely for all methods. A pruned model that looks nearly intact under probability-based evaluation may already be severely damaged

Table 5: Recovery methods applied to the pruned Llama-3.1-8B-Instruct. Scores in percentages. **Pruned**: pruned model before recovery; **SFT**: supervised finetuning; **SGR**: self-distillation from generated data; **GKD**: on-policy self-distillation; **KL**: self-distillation with KL regularization.

Method	MMLU		GSM8K	HumanEval
	Prob.	Gen.		
Unpruned	67.7	67.5	80.7	63.4
Pruned	64.7	64.3	10.6	0.0
SFT	65.5	62.3	61.7	5.5
SGR	66.8	65.5	68.6	11.0
GKD	67.8	66.1	67.6	9.8
KL	66.7	64.6	66.7	**19.5**

Table 6: We use self-distillation with KL regularization to recover the performance of different layer-wise pruning methods. Methods whose generation performance collapses to similar levels after pruning (Table 4) recover to markedly different levels. Scores in percentages.

Pruning method	MMLU		GSM8K	HumanEval
	Prob.	Gen.		
Unpruned	67.7	67.5	80.7	63.4
Reverse	65.1	64.5	66.7	19.5
BI	64.3	63.7	70.8	39.0
PPL	63.2	62.9	63.3	31.7
Taylor	66.0	64.7	14.6	19.5
Random	38.0	0.9	27.1	0.0

in generation capabilities. This shows that the layers that prior work removes are not redundant in general. Instead, they are unused only under probability-based evaluation, and dropping them collapses generation.

5.2 Efficient Recovery via Self-Distillation

Self-distillation has been used to recover model quality after layer-wise pruning (Thangarasa et al., 2025; Shrestha et al., 2026). Both prior works recover with self-distillation from generated data and do not test other distillation methods. We therefore compare four recovery methods. We test supervised finetuning, self-distillation from generated data (also called the SGR method by Shrestha et al. (2026)), self-distillation with KL regularization and on-policy self-distillation using the GKD method (Agarwal et al., 2024). Distillation and finetuning are performed on the Alpaca dataset. Among the four methods, self-distillation with KL regularization recovers the most overall, with the largest gain on HumanEval (Table 5). All self-distillation-based methods out-perform supervised finetuning in recovery.

Whereas previous works often perform self-distillation on reasoning datasets (Thangarasa et al., 2025), the distillation here is performed on a general-purpose corpus. Therefore, it is unlikely that the recovery comes from the model acquiring new reasoning ability. Instead, this shows that layer-wise pruning perturbs the model's output distribution and generation behavior, while much of the underlying capability remains intact and recoverable. The recovery is also lightweight. As shown in Table 5, 500 distillation steps already restore most of the lost GSM8K performance and yield partial recovery on HumanEval.

Furthermore, we notice that different layer-wise pruning methods that collapse to similarly low generation performance right after pruning differ sharply in how much recovery restores (Table 6), with some failing to recover at all. This implies that benchmark scores after pruning may not fully reflect layer importance.

6 Conclusions

In summary, we revisited claims that rank-based and layer-wise pruning can compress large language models while preserving, or even improving, their reasoning ability. Our central finding is that these reported gains are largely an artifact of the evaluation protocol used to select and score the pruned model, rather than evidence of improved reasoning. For rank-based pruning, we gave both a theoretical and an empirical analysis showing that the method removes a surface form. For layer-wise pruning, we show that the method greatly harms generation capabilities in order to preserve parameters. We further showed that the loss in generation capability is recoverable via self-distillation.

This paper challenges the existing narrative that there exists a large amount of redundancy in LLMs that can be removed without harming performance. It provides a concrete case in which these components are not redundant. They are simply unutilized by the evaluation protocol. Although removing these components may seem neutral or even beneficial under the evaluation protocol, removing them still harms the overall reasoning capability.

Limitations

In this paper we only study two pruning methods, rank-based pruning and layer-wise pruning. Although these two methods are representative, there exist many other pruning methods, both structured and unstructured, that are not studied in this paper. These more advanced pruning methods may remove weights in a more careful way, and thus may not be directly interpreted by the analysis in this paper. We encourage future work to study and interpret the potential evaluation bias of these more advanced pruning methods.

Use of AI Assistance

AI-based assistants were used to help set up and refine parts of our experimental pipeline and to polish the writing of this paper. All experimental design choices, results, and claims were verified by the authors, who take full responsibility for the content.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. *Preprint*, arXiv:2306.13649.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. *Preprint*, arXiv:2304.01373.

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. 2023. Birth of a transformer: A memory viewpoint. *Advances in Neural Information Processing Systems*, 36:1560–1588.

Lei Chen, Joan Bruna, and Alberto Bietti. 2025a. Distributional associations vs in-context reasoning: A study of feed-forward and attention layers. In *The Thirteenth International Conference on Learning Representations*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. 2025b. Streamlining redundant layers to compress large language models. *Preprint*, arXiv:2403.19135.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Siqi Fan, Xin Jiang, Xiang Li, Xuying Meng, Peng Han, Shuo Shang, Aixin Sun, Yequan Wang, and Zhongyuan Wang. 2024. Not all layers of llms are necessary during inference. *Preprint*, arXiv:2403.02181.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. *Preprint*, arXiv:2301.00774.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A Roberts. 2025. The unreasonable ineffectiveness of the deeper layers. In *International Conference on Learning Representations*, volume 2025, pages 81906–81920.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. 2024. Shortened llama: Depth pruning for large language models with comparison of retraining methods. *Preprint*, arXiv:2402.02834.

Minkyu Kim, Vincent-Daniel Yun, Youngrae Kim, Youngjin Heo, Suin Cho, Seong hun Kim, Woosang Lim, and Gaeul Kwon. 2026. Rethinking layer redundancy in large language models: Calibration objectives and search for depth pruning. *Preprint*, arXiv:2604.24938.

Yao Lu, Hao Cheng, Yujie Fang, Zeyu Wang, Jiaheng Wei, Dongwei Xu, Qi Xuan, Xiaoniu Yang, and Zhaowei Zhu. 2024. Reassessing layer pruning in llms: New insights and methods. *Preprint*, arXiv:2411.15558.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Preprint*, arXiv:2305.11627.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. 2024. Shortgpt: Layers in large language models are more redundant than you expect. *Preprint*, arXiv:2403.03853.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2023. Locating and editing factual associations in gpt. *Preprint*, arXiv:2202.05262.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Pratyusha Sharma, Jordan T. Ash, and Dipendra Misra. 2024. The truth is in there: Improving reasoning in language models with layer-selective rank reduction. In *The Twelfth International Conference on Learning Representations*.

Safal Shrestha, Anubhav Shrestha, Aadim Nepal, Minwu Kim, and Keith Ross. 2026. On the limits of layer pruning for generative reasoning in large language models. *Preprint*, arXiv:2602.01997.

Mingjie Sun, Zhuang Liu, Anna Bair, and J. Zico Kolter. 2024. A simple and effective pruning approach for large language models. *Preprint*, arXiv:2306.11695.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Vithursan Thangarasa, Ganesh Venkatesh, Mike Lasby, Nish Sinnadurai, and Sean Lie. 2025. Self-data distillation for recovering quality in pruned large language models. *Preprint*, arXiv:2410.09982.

Keyu Wang, Tian Lyu, Guinan Su, Jonas Geiping, Lu Yin, Marco Canini, and Shiwei Liu. 2026. When fewer layers break more chains: Layer pruning harms test-time scaling in LLMs.

Qinsi Wang, Jinghan Ke, Masayoshi Tomizuka, Yiran Chen, Kurt Keutzer, and Chenfeng Xu. 2025. Dobi-svd: Differentiable svd for llm compression and some new perspectives. *Preprint*, arXiv:2502.02723.

Wenxiao Wang, Wei Chen, Yicong Luo, Yongliu Long, Zhengkai Lin, Liye Zhang, Binbin Lin, Deng Cai, and Xiaofei He. 2024a. Model compression and efficient inference for large language models: A survey. *arXiv preprint arXiv:2402.09748*.

Xinpeng Wang, Chengzhi Hu, Bolei Ma, Paul Röttger, and Barbara Plank. 2024b. Look at the text: Instruction-tuned language models are more robust multiple choice selectors than you think. *Preprint*, arXiv:2404.08382.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. *Preprint*, arXiv:2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared llama: Accelerating language model pre-training via structured pruning. In *International Conference on Learning Representations*, volume 2024, pages 5385–5409.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *Preprint*, arXiv:1905.07830.

Mingyang Zhang, Hao Chen, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. 2024. LoRAPrune: Structured pruning meets low-rank parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3013–3026, Bangkok, Thailand. Association for Computational Linguistics.

Xunyu Zhu, Jian Li, Yong Liu, Can Ma, and Weiping Wang. 2024. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12:1556–1577.

A Theoretical Details and Proofs

Model Architecture and Formulation We analyze a one-layer transformer, following theoretical setups of Chen et al. (2025a) and Bietti et al. (2023). The model and data generation process follow Chen et al. (2025a): the architecture and data distribution are similar, but we adopt a different target setup and fewer assumptions to make the setting more realistic when completing the proof. The model takes an input sequence $z_{1:T} = [z_1, \ldots, z_T] \in [N]^T$ over a vocabulary of size N. Two fixed embedding matrices $\mathbf{W}_E \in \mathbb{R}^{N\times d}$ and $\tilde{\mathbf{W}}_E \in \mathbb{R}^{N\times d}$ map tokens into $\mathbb{R}^d$, and a fixed (non-trainable) unembedding matrix $\mathbf{W}_U \in \mathbb{R}^{N\times d}$ maps hidden states back to logits over the vocabulary.

The model consists of a single attention layer and a single feed-forward layer. The attention layer consists of a query-key matrix $\mathbf{W}_{QK}$ and a value matrix $\mathbf{W}_V$. The feed-forward layer is a single linear map $\mathbf{W}_F$ without activation.

Position encoding is represented by the sum of two fixed embeddings. $\mathbf{W}_E$ for the current token z_t and $\tilde{\mathbf{W}}_E$ for the previous token z_{t-1}. This construction follows the previous-token head of Bietti et al. (2023).

Forward Pass Let vocabulary $\mathcal{V} = [N]$, on input $u_{1:m} = (u_1, \ldots, u_m) \in \mathcal{V}^m$, the forward pass is defined as follows. We set $\tilde{\mathbf{W}}_E(u_0) := 0$.

$$x_i \triangleq \mathbf{W}_E(u_i) + \tilde{\mathbf{W}}_E(u_{i-1}) \qquad \in \mathbb{R}^d, \tag{1a}$$

$$a_{1:m} \triangleq \mathrm{softmax}\big(x_m^\top \mathbf{W}_{QK}\, x_{1:m}\big) \qquad \in \Delta^{m-1}, \tag{1b}$$

$$\phi \triangleq \textstyle\sum_{i=1}^m a_i\, \mathbf{W}_V x_i \qquad \in \mathbb{R}^d, \tag{1c}$$

$$\xi_{\text{attn}} \triangleq \mathbf{W}_U\, \phi \qquad \in \mathbb{R}^N, \tag{1d}$$

$$\xi_{\text{ff}} \triangleq \mathbf{W}_U\, \mathbf{W}_F\, x_m \qquad \in \mathbb{R}^N, \tag{1e}$$

$$\xi(u_{1:m}) \triangleq \xi_{\text{attn}} + \xi_{\text{ff}} \qquad \in \mathbb{R}^N, \tag{1f}$$

$$p_\theta(\cdot \mid u_{1:m}) \triangleq \mathrm{softmax}\big(\xi(u_{1:m})\big) \qquad \in \Delta^{N-1}. \tag{1g}$$

Here Δ^{k-1} denotes the k-dimensional probability simplex.

Loss Function Under prompt $z_{1:T}$ and ground truth $y_{1:S} = (y_1, \ldots, y_S) \in \mathcal{V}^*$, let the loss be the autoregressive negative log-likelihood

$$\mathcal{L}\big(z_{1:T},\, y_{1:S}\big) \triangleq \sum_{s=1}^{S} -\log p_\theta\big(y_s \,\big|\, (z_{1:T},\, y_{1:s-1})\big), \tag{2}$$

Training Data. The input $z_{1:T}$ is generated as a Markov prompt process. Let $\mathcal{V} = [N]$ and fix a special trigger token $q \in \mathcal{V}$. For each example, draw a latent answer token

$$\bar{y} \sim \mathrm{Unif}\big(\mathcal{V} \setminus (\{q\} \cup \{\tau_1, \ldots, \tau_L\})\big).$$

The prompt is then sampled from a Markov chain in which the trigger deterministically emits the answer token and every other state transitions uniformly:

$$z_1 \sim \mathrm{Unif}(\mathcal{V} \setminus \{q\})$$

$$z_{t+1} \mid z_t, \bar{y} \sim \begin{cases} \delta_{\bar{y}}, & z_t = q, \\ \mathrm{Unif}(\mathcal{V} \setminus \{q\}), & z_t \neq q, \end{cases}$$

for $t = 1, \ldots, T-2$. We also require $z_T = q$.

Target We consider a simple binary output-formatting scheme to be the target. Fix a prefix $(\tau_1, \ldots, \tau_L)$, $f(\bar{y}) = (\tau_1, \ldots, \tau_L, \bar{y})$. For each sample with answer token $\bar{y}$, the target sequence independently

$$y_{1:S} = \begin{cases} (\tau_1, \ldots, \tau_L, \bar{y}), & \text{with probability } \alpha, \\ (\bar{y}), & \text{with probability } 1-\alpha, \end{cases} \tag{3}$$

Standing Assumptions

(A1) **Training conditions.** We consider the vocabulary size N to be sufficiently large at $N \to \infty$. The model is trained by gradient descent from zero initialization at $W_V = W_F = W_{QK} = 0$. The learning rate is $\eta = O(1)$. The dimension of the model matrices is $d \geq 2N$. The batch size $m = \Omega(\log^2 N)$ and sequence length $T = \Omega(\log N)$.

(A2) **Fixed orthogonal embeddings.** The embedding matrices $W_E, \tilde{W}_E \in \mathbb{R}^{N\times d}$ are frozen and satisfy, for all $i, j \in [N]$,

$$\langle W_E(i), W_E(j)\rangle = \delta_{ij},$$
$$\langle \tilde{W}_E(i), \tilde{W}_E(j)\rangle = \delta_{ij},$$
$$\langle W_E(i), \tilde{W}_E(j)\rangle = 0.$$

Consequently, for $x_t = W_E(z_t) + \tilde{W}_E(z_{t-1})$, we have $\|x_t\|_2 = \sqrt{2}$.

(A3) **Fixed orthogonal unembedding.** The unembedding matrix $W_U \in \mathbb{R}^{N\times d}$ is frozen and satisfies, for all $i, j \in [N]$,

$$W_U(i)^\top W_E(j) = \delta_{ij},$$
$$W_U(i)^\top \tilde{W}_E(j) = 0.$$

Writing $\mathcal{S}$ for the $2N$-dimensional subspace spanned by all embedding rows $\{W_E(i), \tilde{W}_E(i)\}_{i\in[N]}$, we further require $W_U v = 0$ for every $v \in \mathcal{S}^\perp$.

A.1 Training Dynamics of FFN

Lemma 1 (Stability of Training for the Attention Layer)**.** *Under the given model and assumptions, let $K = \Theta(\log N)$, with probability at least $1-\delta$, simultaneously for all $s \le K$, $j, k \in [N]$,*

$$\left|\mathbf{W}_U(j)^\top \big(\nabla_{\mathbf{W}_V} \hat{L}^{(s)}\big)\mathbf{W}_E(k) - \mu^{(s)}(j,k)\right| \le O\left(\sqrt{\frac{\log(NK/\delta)}{m}}\right). \tag{4}$$

with

$$\mu^{(s)}(j,k) = O\Big(\tfrac{1}{\log N}\Big),$$
$$\|W_{QK}^{(s)} - W_{QK}^{(0)}\|_F = O(1).$$

Lemma 2 (Stability of Training for the Feed-Forward Layer)**.** *Under the given model and assumptions, with probability at least $1-\delta$, simultaneously for all $s \le K$, $k \in [N]$,*

$$\left|\mathbf{W}_U(k)^\top \big(\nabla_{W_F} \hat{L}^{(s)}\big)\mathbf{W}_E(q) - \mu_q^{(s)}(k)\right| \le O\left(\sqrt{\frac{\log(NK/\delta)}{m}}\right).$$

Further,

$$\mu_q^{(s)}(\tau) = \mathbb{E}[p_\tau^{(s,q)}] - \alpha,$$
$$\mu_q^{(s)}(k) = \mathbb{E}[p_k^{(s,q)}] + O\big(\tfrac{1}{N}\big) \quad (k \ne \tau).$$

A.2 Proofs

Proof of Lemma 1.

Proof. We denote the per-position cross-entropy by

$$\ell(\xi, y) := -\log \frac{\exp(\xi_y)}{\sum_{j\in\mathcal{V}} \exp(\xi_j)}.$$

The loss function (2) is a sum of $L+1$ fixed-target prediction terms over the transitions $q \to \tau_1 \to \cdots \to \tau_L \to \bar{y}$. Since the prefix is deterministic, every transition $\tau_{\ell-1} \to \tau_\ell$ has the same form as $q \to \tau_1$, so without loss of generality we analyze $q \to \tau_1$ only.

Let $Z = (z_{1:T}, y_{1:S})$ denote a single training batch consisting of samples $Z_1, \dots, Z_m$. For classes $j, k \in [N]$, define the (j,k) value-gradient channel

$$g_{j,k}^{(s)}(Z) := \mathbf{W}_U(j)^\top \big(\nabla_{W_V} \ell(\xi^{(s)}(Z), y)\big)\mathbf{W}_E(k),$$

Its batch average and population mean are

$$\widehat{g}_{j,k}^{(s)} := \tfrac{1}{m}\sum_{i=1}^{m} g_{j,k}^{(s)}(Z_i),$$
$$\mu^{(s)}(j,k) := \mathbb{E}[g_{j,k}^{(s)}(Z)],$$

Step 1: concentration for $\widehat{g}_{j,k}^{(s)}$. We have,

$$\nabla_{W_V}\ell = (\nabla_\phi \ell)\,\bar{x}^\top,$$
$$\bar{x} := \sum_{t\le T} a_t x_t,$$
$$\nabla_\phi \ell = W_U^\top (p - e_y).$$

By $(A2)$ and $(A3)$: $\|x_t\|_2 = \sqrt{2}$, $\|W_U\|_{\mathrm{op}} \le 1$, and $\|p - e_y\|_2 \le \sqrt{2}$. Hence $\|\nabla_\phi \ell\|_2 \le \sqrt{2}$, $\|\bar{x}\|_2 \le \sqrt{2}$, and therefore $\|\nabla_{W_V}\ell\|_F \le 2$. With $\|W_E(k)\|_2 = 1$, $\|W_U(j)\|_2 \le 1$, we get $|g_{j,k}^{(s)}(Z)| \le 2$ almost surely.

For a given s, the Hoeffding bound gives

$$\Pr\Big(\big|\widehat{g}_{j,k}^{(s)} - \mu^{(s)}(j,k)\big| \ge \epsilon\Big) \le 2\exp\Big(-\frac{m\epsilon^2}{8}\Big).$$

A union bound over $s \in \{0, \dots, K\}$ and $(j,k) \in [N]^2$ yields

$$\left|\mathbf{W}_U(j)^\top \left(\nabla_{\mathbf{W}_V} \hat{\mathbf{L}}^{(s)}\right)\mathbf{W}_E(k) - \mu^{(s)}(j,k)\right| \le O\left(\sqrt{\frac{\log(NK/\delta)}{m}}\right),$$

simultaneously for all $s \le K$, $j, k \in [N]$, with probability at least $1 - \delta/2$. We call this event E_1.

Step 2: deterministic induction over four invariants. Assume that the condition of E_1 holds. Let $K = \kappa \log N$ with fixed $\kappa > 0$. We use:

$$(A_s): \quad \max_{r\le s} \|W_{QK}^{(r)} - W_{QK}^{(0)}\|_F \le C_Q,$$

$$(B_s): \quad \max_{r\le s}\max_{j,k\in[N]} |\mu^{(r)}(j,k)| \le \frac{C_\mu}{\log N}.$$

$$(C_s): \quad \max_{r\le s} \left\| \mathbb{E}\left[\nabla_{W_{QK}} \ell^{(r)}\right] \right\|_F \le \frac{C_G}{\log N}.$$

$$(D_s): \quad \max_{r\le s} \|W_V^{(r)} - W_V^{(0)}\|_{\mathrm{op}} \le C_V.$$

Here C_Q, C_μ, C_G, C_V are constants that are independent from N.

Base case. At initialization $W_{QK}^{(0)} = 0$, (A_0) holds. Also $\xi^{(0)} \equiv 0$, hence $a_t^{(0)} = 1/T$. All channels are even so $|\mu^{(0)}(j,k)| = O(1/N) = O(1/\log N)$, which gives (B_0) holds.

$$\nabla_{W_{QK}} \ell^{(0)} = \sum_{t\le T} \beta_t^{(0)} x_T x_t^\top,$$
$$\beta_t^{(0)} = a_t^{(0)} \langle g^{(0)}, v_t^{(0)} - \phi^{(0)} \rangle.$$

Since $W_V^{(0)} = 0$, we have $v_t^{(0)} = \phi^{(0)} = 0$, hence $\nabla_{W_{QK}} \ell^{(0)} = 0$, so (C_0) holds. $\|W_V^{(0)} - W_V^{(0)}\|_{\mathrm{op}} = 0 \le C_V$, so (D_0) holds.

Inductive step I: $(A_s, B_s, D_s) \Rightarrow (C_{s+1})$**.** Fix $r \le s < K$. Define scores

$$S_t^{(r)} := x_T^\top W_{QK}^{(r)} x_t,$$
$$a_t^{(r)} = \frac{\exp(S_t^{(r)})}{\sum_{u\le T} \exp(S_u^{(r)})}.$$

From (A_s) and $\|x_t\|_2 = \sqrt{2}$, $|S_t^{(r)}| \le 2C_Q$, so for some constant C_a,

$$\frac{e^{-2C_Q}}{T} \le a_t^{(r)} \le \frac{e^{2C_Q}}{T}$$
$$a_t^{(r)} \le \frac{C_a}{\log N}.$$

Moreover the W_{QK}-gradient has form

$$\nabla_{W_{QK}} \ell^{(r)} = \sum_{t\le T} \beta_t^{(r)}\, x_T x_t^\top,$$
$$\beta_t^{(r)} := a_t^{(r)} \langle g^{(r)}, v_t^{(r)} - \phi^{(r)} \rangle,$$

where $g^{(r)} = W_U^\top (p^{(r)} - e_y)$, $v_t^{(r)} = W_V^{(r)} x_t$, and $\phi^{(r)} = \sum_{u\le T} a_u^{(r)} v_u^{(r)}$.

Set

$$\psi_t^{(r)} := \langle g^{(r)}, v_t^{(r)} - \phi^{(r)} \rangle,$$
$$\beta_t^{(r)} = a_t^{(r)} \psi_t^{(r)}.$$

By Cauchy–Schwarz, $|\psi_t^{(r)}| \le \|g^{(r)}\|_2 \|v_t^{(r)} - \phi^{(r)}\|_2$. By $(A2)$–$(A3)$, $\|g^{(r)}\|_2 = \|W_U^\top (p^{(r)} - e_y)\|_2 \le \sqrt{2}$; and since $v_t^{(r)} = W_V^{(r)} x_t$ with $\|x_t\|_2 = \sqrt{2}$ and $\phi^{(r)}$ a convex average of the $v_u^{(r)}$, invariant (D_s) gives $\|v_t^{(r)} - \phi^{(r)}\|_2 \le 2\sqrt{2}\, \|W_V^{(r)}\|_{\mathrm{op}} \le 2\sqrt{2}\, C_V$.

$$|\psi_t^{(r)}| \le 4C_V =: C_\psi,$$

where C_ψ is a N-independent constant. Since the attention scores are $O(1)$, the softmax is within a constant factor of uniform, so $a_t^{(r)} = \Theta(1/T) = O(1/\log N)$.

Consider the per-position $\psi_t^{(r)}$, it is updated only when $\{z_t = k\}$. Because z_t is uniform, $\Pr(z_t = k \mid z_T = q, \bar{y}) \le 1/(N-1)$. With $|\psi_t^{(r)}| \le C_\psi$ from above, we have

$$\mathbb{E}\left[|\psi_t^{(r)}|\right] \le \frac{C_\psi}{N-1} = O\left(\frac{1}{N}\right).$$

$$\sum_{t\le T} \mathbb{E}\left[|\psi_t^{(r)}|\right] = O\left(\frac{T}{N}\right).$$

Combine this with $a_t^{(r)} \le C_a/\log N$,

$$\mathbb{E}\left[|\beta_t^{(r)}|\right] \le \frac{C_a}{\log N}\, \mathbb{E}\left[|\psi_t^{(r)}|\right] = O\left(\frac{1}{N\log N}\right).$$

$$\begin{aligned}\left\| \mathbb{E}\left[\nabla_{W_{QK}} \ell^{(r)}\right] \right\|_F &\le \sum_{t\le T} \mathbb{E}\left[|\beta_t^{(r)}|\right] \|x_T x_t^\top\|_F \\ &= O\left(\tfrac{T}{N\log N}\right) = O\left(\tfrac{1}{\log N}\right),\end{aligned}$$

this gives (C_{s+1}).

Inductive step II: $(C_{s+1}) \Rightarrow (A_{s+1})$**.** Weights of W_{QK} are updated with gradient descent

$$\Delta W_{QK}^{(r)} := W_{QK}^{(r+1)} - W_{QK}^{(r)} = -\eta\, \nabla_{W_{QK}} \hat{L}^{(r)}.$$

Denote the event E_2 for all $r \le K$,

$$\begin{aligned}&\left\|\nabla_{W_{QK}} \hat{L}^{(r)} - \mathbb{E}[\nabla_{W_{QK}} \ell^{(r)}]\right\|_F \\ &\quad + \left\|\nabla_{W_V} \hat{L}^{(r)} - \mathbb{E}[\nabla_{W_V} \ell^{(r)}]\right\|_{\mathrm{op}} \le O\left(\tfrac{1}{\log N}\right),\end{aligned}$$

By a similar Hoeffding bound of Step 1, this holds with probability at least $1 - \delta/2$. By a union bound over E_1 and E_2, with $1 - \delta$ the proof holds. Under this

$$\left\|\nabla_{W_{QK}} \hat{L}^{(r)} - \mathbb{E}[\nabla_{W_{QK}} \ell^{(r)}]\right\|_F \le O\left(\frac{1}{\log N}\right)$$

for all $r \le K$. Combining with (C_{s+1}),

$$\|\nabla_{W_{QK}} \hat{L}^{(r)}\|_F = O\left(\frac{1}{\log N}\right)$$

for all $r \le K$. Therefore

$$\|\Delta W_{QK}^{(r)}\|_F \le \eta \, \|\nabla_{W_{QK}} \hat{L}^{(r)}\|_F = O\Big(\tfrac{1}{\log N}\Big).$$

$$\begin{aligned}\|W_{QK}^{(s+1)} - W_{QK}^{(0)}\|_F &\le \sum_{r=0}^{s} \|\Delta W_{QK}^{(r)}\|_F \\ &= K\, O\Big(\tfrac{1}{\log N}\Big) = O(1),\end{aligned}$$

this gives (A_{s+1}).

Inductive step III: $(A_{s+1}, C_{s+1}) \Rightarrow (B_{s+1})$**.** For $r \le s$, define

$$\Delta\mu^{(r)}(j,k) := \mu^{(r+1)}(j,k) - \mu^{(r)}(j,k). \quad (5)$$

We show that

$$\begin{aligned}\Big|\Delta\mu^{(r)}(j,k)\Big| &\le \frac{L_\mu}{\log N} \, \|\Delta W_{QK}^{(r)}\|_F \\ &\quad + \frac{L_V}{N} \, \|\Delta W_V^{(r)}\|_{\mathrm{op}},\end{aligned}$$

For a fixed j, k define

$$F_{j,k}(W_{QK}, W_V) := \mathbf{W}_U(j)^\top \big(\nabla_{W_V} \ell\big) \mathbf{W}_E(k)$$

as a function of the pair (W_{QK}, W_V), so by the mean value theorem,

$$\begin{aligned}|\Delta\mu^{(r)}(j,k)| &\le \mathbb{E}\big[\sup \|\partial_{W_{QK}} F_{j,k}\|\big] \, \|\Delta W_{QK}^{(r)}\|_F \\ &\quad + \mathbb{E}[\sup \|\partial_{W_V} F_{j,k}\|] \, \|\Delta W_V^{(r)}\|_{\mathrm{op}}.\end{aligned}$$

We bound each term individually.

W_{QK} term. Apply the chain rule $W_{QK} \to S \to a \to F_{j,k}$. The score-to-attention Jacobian is the standard softmax derivative $\partial a_u / \partial S_t = a_u(\delta_{ut} - a_t)$, and $\partial S_t / \partial W_{QK} = x_T x_t^\top$.

$$\begin{aligned}\frac{\partial F_{j,k}}{\partial W_{QK}} &= \sum_t \frac{\partial F_{j,k}}{\partial S_t} \, x_T x_t^\top \\ &= \sum_u \frac{\partial F_{j,k}}{\partial a_u} \Big(\sum_t \frac{\partial a_u}{\partial S_t} \, x_T x_t^\top \Big).\end{aligned}$$

Define $\alpha := W_U(j)^\top W_U^\top (p - e_y)$ and $\beta := \sum_v a_v \mathbf{1}\{z_v = k\}$,

$$F_{j,k} = \alpha\beta$$

$$\begin{aligned}\frac{\partial F_{j,k}}{\partial a_u} &= \beta \, W_U(j)^\top W_U^\top (\mathrm{diag}(p) - pp^\top) W_U W_V x_u \\ &\quad + \alpha \, \mathbf{1}\{z_u = k\}.\end{aligned}$$

Using $\partial a_u / \partial S_t = a_u(\delta_{ut} - a_t)$:

$$\sum_t \frac{\partial a_u}{\partial S_t} \, x_t = a_u(x_u - \bar{x}),$$

$$\Big\| \sum_t \frac{\partial a_u}{\partial S_t} \, x_T x_t^\top \Big\|_F \le \sqrt{2} \, a_u \, \|x_u - \bar{x}\|_2 = O(1/T).$$

With $|\partial F_{j,k} / \partial a_u| = O(1)$,

$$\Big\| \frac{\partial F_{j,k}}{\partial W_{QK}} \Big\|_F \le \sum_u \Big| \frac{\partial F_{j,k}}{\partial a_u} \Big| \cdot O(1/T) = O(1).$$

With $\mathbb{E}[\mathbf{1}\{z_u = k\}], \mathbb{E}[|\beta|] \le 1/(N-1)$ and $\mathbb{E}[|\partial F_{j,k} / \partial a_u|] = O(1/N)$:

$$\begin{aligned}\mathbb{E}[\sup \|\partial_{W_{QK}} F_{j,k}\|_F] &\le L_\mu / N = O(1/N) \\ &\subseteq O(1/\log N).\end{aligned}$$

W_V term. Apply the chain rule $W_V \to \phi \to \xi \to p \to \nabla_\phi \ell \to F_{j,k}$. With $F_{j,k} = \alpha\beta$, the factor $\beta = \sum_t a_t \mathbf{1}\{z_t = k\}$ has no W_V dependence, so $\phi = W_V \bar{x}$ is linear in W_V,

$$\begin{aligned}\frac{\partial F_{j,k}}{\partial W_V} &= \beta \cdot \frac{\partial \alpha}{\partial W_V} = \beta \cdot M \, \bar{x}^\top, \\ M &:= W_U^\top (\mathrm{diag}(p) - pp^\top) W_U W_U(j),\end{aligned}$$

M is rank 1, so

$$\Big\| \frac{\partial F_{j,k}}{\partial W_V} \Big\|_{\mathrm{op}} \le |\beta| \cdot \|M\|_2 \cdot \|\bar{x}\|_2 \le \sqrt{2} \, |\beta|,$$

using $\|\mathrm{diag}(p) - pp^\top\|_{\mathrm{op}} \le 1$, $\|W_U\|_{\mathrm{op}} \le 1$, and $\|\bar{x}\|_2 \le \sqrt{2}$.

$$\mathbb{E}[|\beta|] \le \sum_t a_t \, \Pr(z_t = k \mid z_T = q, \bar{y}) \le \frac{1}{N-1},$$

so

$$\mathbb{E}[\sup \|\partial_{W_V} F_{j,k}\|_{\mathrm{op}}] \le \frac{L_V}{N},$$

for an N-independent L_V.

Thus, for $s + 1 \le K$,

$$\begin{aligned}|\mu^{(s+1)}(j,k)| &\le |\mu^{(0)}(j,k)| + \sum_{r=0}^{s} |\Delta\mu^{(r)}(j,k)| \\ &\le O\Big(\tfrac{1}{\log N}\Big) + K\, O\Big(\tfrac{1}{\log^2 N}\Big) \\ &= O\Big(\tfrac{1}{\log N}\Big),\end{aligned}$$

which is (B_{s+1}).

Inductive step IV: $(B_{s+1}) \Rightarrow (D_{s+1})$**.** Weights of W_V are updated with gradient descent $W_V^{(r+1)} - W_V^{(r)} = -\eta \, \nabla_{W_V} \hat{L}^{(r)}$. By $(A2)$–$(A3)$, the operator norm of the mean gradient is bounded by its largest (j,k) channel: $\|\mathbb{E}[\nabla_{W_V} \ell^{(r)}]\|_{\mathrm{op}} \le C \max_{j,k} |\mu^{(r)}(j,k)| = O(1/\log N)$ by (B_{s+1}); and E_2 bounds the sampling deviation by

$o(1/\log N)$. Each step therefore moves W_V by $O(1/\log N)$ in operator norm, and over $K = \Theta(\log N)$ steps,

$$\|W_V^{(s+1)} - W_V^{(0)}\|_{\mathrm{op}} \le K \cdot \eta \cdot O(1/\log N) = O(1) \le C_V,$$

which is (D_{s+1}).

Hence (A_s), (B_s), (C_s), and (D_s) hold for all $s \le K$. This gives

$$\mu^{(s)}(j,k) = O\Big(\tfrac{1}{\log N}\Big),$$
$$\|W_{QK}^{(s)} - W_{QK}^{(0)}\|_F = O(1),$$
$$\|W_V^{(s)} - W_V^{(0)}\|_{\mathrm{op}} = O(1),$$

for all $s \le K$. For a constant number of additional steps after K, the steps create only $O(1)$ change. Therefore the theorem follows when E_1 and E_2 both hold, which is at probability $1-\delta$. □

Proof of Lemma 2.

Proof. Similar to the proof of Lemma 1, we only consider $q \to \tau_1$ without loss of generality.

Let $Z = (z_{1:T}, y_{1:S})$ denote a single training batch consisting of samples $Z_1, \dots, Z_m$. For class $k \in [N]$, define the k-th FFN-gradient channel

$$h_k^{(s)}(Z) \;:=\; \mathbf{W}_U(k)^\top \big(\nabla_{W_F} \ell(\xi^{(s)}(Z), y)\big) \mathbf{W}_E(q),$$

Its batch average and population mean are

$$\widehat{h}_k^{(s)} := \tfrac{1}{m} \sum_{i=1}^{m} h_k^{(s)}(Z_i),$$
$$\mu_q^{(s)}(k) := \mathbb{E}[h_k^{(s)}(Z)].$$

By (A2)–(A3), $\langle x_t, W_E(q)\rangle = \mathbf{1}\{z_t = q\}$, so the projection $\nabla_{W_F}\ell \cdot W_E(q)$ vanishes at every prediction position except $t = T$, where $z_T = q$ by construction. The FFN gradient at that one position is $W_U^\top (p - e_{y_1})\, x_T^\top$, and applying $W_U(k)^\top W_U^\top = e_k^\top$ from (A3) on the left gives

$$h_k^{(s)}(Z) = p_k^{(s,q)} - \mathbf{1}\{y_1 = k\},$$

where $p^{(s,q)}$ is the model softmax at the query position and y_1 is the first supervised token. Hence $|h_k^{(s)}(Z)| \le 1$ almost surely.

For a given s, the Hoeffding bound gives

$$\Pr\Big(\Big|\widehat{h}_k^{(s)} - \mu_q^{(s)}(k)\Big| \ge \epsilon\Big) \le 2\exp\Big(-\frac{m\epsilon^2}{2}\Big).$$

A union bound over $s \in \{0, \dots, K\}$ and $k \in [N]$ yields

$$\big|\widehat{h}_k^{(s)} - \mu_q^{(s)}(k)\big| \le O\left(\sqrt{\frac{\log(NK/\delta)}{m}}\right),$$

simultaneously for all $s \le K$, $k \in [N]$, with probability at least $1-\delta$.

Taking expectation in $h_k^{(s)} = p_k^{(s,q)} - \mathbf{1}\{y_1 = k\}$,

$$\mu_q^{(s)}(k) = \mathbb{E}[p_k^{(s,q)}] - \Pr(y_1 = k \mid z_T = q).$$

Under the target generation process, the transition $q \to \tau_1$ has probability α, each subsequent prefix transition $\tau_{\ell-1} \to \tau_\ell$ has probability 1, and any other target token at the query position has probability $O(1/N)$. In particular $\Pr(y_1 = \tau_1 \mid z_T = q) = \alpha$ and $\Pr(y_1 = k \mid z_T = q) = O(1/N)$ for $k \ne \tau_1$. Therefore

$$\mu_q^{(s)}(\tau) = \mathbb{E}[p_\tau^{(s,q)}] - \alpha,$$
$$\mu_q^{(s)}(k) = \mathbb{E}[p_k^{(s,q)}] + O\big(\tfrac{1}{N}\big) \quad (k \ne \tau),$$

which proves the lemma. □

Proof of Theorem 1.

Proof. Apply Lemmas 1 and 2 each with failure probability $\delta/2$ and take a union bound; condition on the intersection of their events, which holds with probability at least $1-\delta$.

For $z_T = q$, define

$$u_{jq}^{(s)} := W_U(j)^\top W_F^{(s)} W_E(q),$$
$$g_j^{(s)} := W_U(j)^\top (\nabla_{W_F} \hat{L}^{(s)}) W_E(q).$$

SGD on W_F gives $u_{jq}^{(s+1)} - u_{jq}^{(s)} = -\eta g_j^{(s)}$, and Lemma 2 yields uniformly for $s \le K$

$$g_\tau^{(s)} = \big(\mathbb{E}[p_\tau^{(s,q)}] - \alpha\big) + o(1),$$
$$g_j^{(s)} = \mathbb{E}[p_j^{(s,q)}] + O(1/N) + o(1) \quad (j \ne \tau),$$

with $o(1)$ the Lemma 2 sampling error ($m = \Omega(\log^2 N)$).

For $j \ne \tau$, $\mathbb{E}[p_j^{(s,q)}] = O(1/N)$ (uniform $\bar{y}$) gives

$$|u_{jq}^{(s+1)} - u_{jq}^{(s)}| = O(\eta/N),$$
$$u_{jq}^{(s)} = O(\eta s/N) = o(1)$$

for $s \le K = \Theta(\log N)$. By Lemma 1 we know the contribution of the attention part is O(1), so we only need to consider the FFN contribution.

Moreover, because of a softmax of N logits, we need $u_{\tau q} = \Theta(\log N)$ for $p_\tau = \Theta(1)$. While $\mathbb{E}[p_\tau^{(s,q)}] \le \alpha/2$,

$$u_{\tau q}^{(s+1)} - u_{\tau q}^{(s)} \ge \eta\alpha/2 - o(\eta) = \Theta(\eta),$$

so

$$s_\star := \min\{s : \mathbb{E}[p_\tau^{(s,q)}] \ge \alpha/2\} = \Theta(\log N).$$

Afterwards, in $O(1)$ steps the $q \to \tau_1$ transition reaches $\mathbb{E}[p_{\tau_1}^{(s,q)}] = \alpha + o(1)$. For all other transitions $\tau_{\ell-1} \to \tau_\ell$ the prefix is deterministic, so the same process tracks the target 1.

Conditioned on the events of Lemmas 1 and 2, the sampling error is $o(1)$ ($m = \Omega(\log^2 N)$). Writing $p^\star(\cdot \mid \cdot)$ for the model's transition probabilities at step $s_\star$,

$$\begin{aligned} p^\star(\tau_1 \mid q) &= \alpha + o(1), \\ p^\star(\tau_\ell \mid \tau_{\ell-1}) &= 1 - o(1) \quad (2 \le \ell \le L). \end{aligned}$$

The format-compliance probability factorizes over these transitions, so for fixed L,

$$\begin{aligned} &p_\theta\big((\tau_1, \ldots, \tau_L) \mid z_T = q\big) \\ &\quad = p^\star(\tau_1 \mid q) \prod_{\ell=2}^{L} p^\star(\tau_\ell \mid \tau_{\ell-1}) \\ &\quad = \big(\alpha + o(1)\big)\big(1 - o(1)\big)^{L-1} \\ &\quad = \alpha + o(1). \end{aligned}$$

Therefore

$$\big|\, p_\theta\big((\tau_1, \ldots, \tau_L) \mid z_T = q\big) - \alpha \,\big| = o(1),$$

which is the relaxed format-compliance bound of Theorem 1(ii).

Lemma 2 shows that the $q \to \tau_1$ transition contributes a rank-1 update, and each subsequent transition $\tau_{\ell-1} \to \tau_\ell$ likewise contributes a rank-1 update to W_F. Summing,

$$W_F^{(s_\star)} = \sum_{\ell=1}^{L} \lambda_\ell \, u_{\tau_\ell} e_{\tau_{\ell-1}}^\top + R^{(s_\star)},$$

with each $\lambda_\ell = \Theta(\log N)$. The residual R for each is $o(1)$ for the first transition and 0 for the other transitions. Therefore, this gives an $o(1)$ residual collectively. □

Table 7: Rank-k reconstruction of the format update ΔW (layer-23 down-projection). Scores are percentages (%).

Reconstruction	Format output rate	Accuracy
W' (clean, no format)	25.8	69.2
$W' + \text{rank-1}(\Delta W)$	61.6	37.6
$W' + \text{rank-2}(\Delta W)$	57.4	41.6
$W' + \text{rank-3}(\Delta W)$	57.8	41.2
$W' + \text{rank-4}(\Delta W)$	57.8	41.2
$W' + \text{rank-5}(\Delta W)$	58.0	41.2
W (full injection)	57.8	41.2

B Further Experiments

B.1 Format Component Is Low-Rank

Beyond being subspace-unaligned (§3.2), the format update ΔW of the layer-23 down-projection is *effectively low-rank* in its effect on the output. Because ΔW is obtained through full finetuning, it is a dense matrix with no explicit low-rank structure. We nonetheless replace the weights with $W' + \text{rank-}k(\Delta W)$ and measure the format output rate and task accuracy (Table 7). A rank-2 component of ΔW already reproduces almost the full format effect of the entire matrix, and higher ranks add no further effect. This shows that ΔW is sufficiently low-rank, corresponding to the low-rank format localization of Theorem 1(i).

All results reported in §3.2 come from a single finetuning-and-pruning run on a fixed seed; we do not average over multiple seeds.

B.2 Rank-Based Pruning Procedure

Rank-based pruning is configured on a validation set $\mathcal{D}_{\text{val}}$ and evaluated on a held-out test set $\mathcal{D}$. A *pruning configuration* is a triple $c = (\ell, t, \rho)$ consisting of a layer ℓ, a matrix type t (the FFN up- or down-projection, the attention projections, etc.), and a preserved-rank ratio ρ. Applying c truncates the SVD of the matrix of type t at layer ℓ to its top $\lceil \rho r \rceil$ singular components (Section 2), yielding a pruned model M_c.

The method proceeds in two steps. First, it iterates over the configuration space and picks the configuration that maximizes accuracy on the validation set,

$$c^\star = \arg\max_{c=(\ell,t,\rho)} \mathrm{Acc}_{\mathcal{D}_{\text{val}}}(M_c).$$

Second, it prunes the model with this best configuration $c^\star$ and evaluates the resulting model $M_{c^\star}$ on the test set $\mathcal{D}$, which is the reported number.

Because $\mathcal{D}_{\text{val}}$ is chosen to share the output format of $\mathcal{D}$, the configuration that best suppresses format on $\mathcal{D}_{\text{val}}$ transfers to $\mathcal{D}$.

In our experiments we fix $\rho = 0.1$ and the matrix type t to the up-projection matrix, which is found to be the most effective choice (Sharma et al., 2024).

B.3 Details of the Evaluation Setup

B.3.1 Rank-based pruning: closed-form evaluation

For each benchmark in Table 2, we use 500 samples as the validation set and 1500 disjoint samples as the test set. We use the model gpt-4.1-mini to generate the candidate tokens for the WikiText dataset. Ten incorrect but confusing candidates are generated for each question, and during evaluation, we extract the one among the 11 candidates (including the correct answer) with the highest probability as the model's answer.

For the CounterFact dataset, because it already provides a false answer, we use the two candidates.

Table 8 reports the pruning configuration selected under this protocol for each model and benchmark—the layer chosen on the validation set (with $\rho = 0.1$ on the `mlp.up` matrix, following Appendix B.2).

Table 8: Pruning is on layer number *Layer* with $\rho = 0.1$ on the up projection matrix.

Model	Benchmark	Layer
GPT-J-6B	MMLU	14
	ARC-Easy	16
	WikiText	27
	CounterFact	27
Llama-2-7B	MMLU	31
	ARC-Easy	31
	WikiText	27
	CounterFact	29
Qwen-2.5-7B-Inst	MMLU	21
	ARC-Easy	20
	WikiText	17
	CounterFact	19
Llama-3.1-8B-Inst	MMLU	19
	ARC-Easy	23
	WikiText	31
	CounterFact	31

For evaluating MMLU and ARC-Easy using generation-based evaluation, we use the following prompts. The same template feeds the probability-based and generation-based variants; the generation variant takes the $\arg\max$ of the next token over the full vocabulary, scored correct when that token equals the gold option letter (the first token of A/B/C/D).

MMLU:

```
The following is a multiple-choice
question. Answer with a single letter
(A, B, C, or D).
Question: {question}
A. {choice_a}
B. {choice_b}
C. {choice_c}
D. {choice_d}
Answer:
```

ARC-Easy:

```
The following is a multiple-choice
question. Answer with a single letter.
Question: {question}
A. {choice_a}
B. {choice_b}
C. {choice_c}
D. {choice_d}
Answer:
```

Admittedly, this may not be the best possible prompt for Llama-2-7B; our goal, however, is not to elicit the best answer from the model, but to demonstrate that the gap between surface form and answer selection can be exploited under certain benchmark settings.

B.3.2 Rank-based pruning: free-form evaluation

We select 300 and 30 samples from the benchmarks as validation set, with random indices generated with seed 512. All other samples belong to the test set. The layer selected for each model and benchmark is reported in Table 9.

Table 9: Layer selected under free-form generation evaluation. Pruning is on layer number *Layer* with $\rho = 0.1$ on the up projection matrix.

Model	Benchmark	Layer
Llama-3.1-8B-Inst	GSM8K	24
	HumanEval	3
Qwen-2.5-7B-Inst	GSM8K	3
	HumanEval	4

B.4 Detailed Statistics

Table 10 reports the full per-model, per-benchmark statistics underlying Table 2.

B.5 Layer-Wise Pruning and Recovery

For the layer-wise experiments of Section 5, we prune 20% of the transformer blocks (6 of 32) of Llama-3.1-8B-Instruct. For recovery, we fine-tune

Table 10: Full per-model, per-benchmark statistics underlying Table 2. Pruning uses preserved-rank ratio $\rho = 0.1$ on the `mlp.up` weight at the layer selected for each benchmark (Table 8). *Base* is the unpruned accuracy and Δ the post-pruning change, both in percentage points. *Raw*: next-token generation accuracy; *Invalid*: invalid-output rate; *Real*: accuracy conditioned on valid outputs; *Prob.*: probability-based (answer-ranking) accuracy. A bold Δ marks a change whose 95% bootstrap CI excludes zero.

Model	Benchmark	Raw		Invalid		Real		Prob.	
		Base	Δ	Base	Δ	Base	Δ	Base	Δ
GPT-J-6B	MMLU	25.7	−1.8	1.9	**+1.8**	26.2	−1.4	26.4	−1.2
	ARC-Easy	25.2	−0.1	0.2	−0.1	25.2	−0.1	25.1	−0.1
	WikiText	33.0	**+10.9**	62.7	**−14.4**	88.5	**−3.6**	78.4	+0.3
	CounterFact	13.3	**+10.1**	86.3	**−10.5**	97.1	−0.1	84.3	+0.3
Llama-2-7B	MMLU	14.8	**+15.7**	70.5	**−61.5**	50.1	**−16.6**	40.3	**−6.7**
	ARC-Easy	10.8	**+31.1**	87.1	**−75.9**	83.5	**−36.4**	59.5	**−13.2**
	WikiText	50.5	**+2.2**	43.6	**−2.7**	89.6	−0.3	82.2	+0.3
	CounterFact	21.9	**+1.3**	77.5	**−1.5**	97.6	−0.7	87.3	−0.1
Qwen-2.5-7B-Inst	MMLU	71.6	−0.3	0.0	+0.1	71.6	−0.3	71.4	+0.0
	ARC-Easy	96.1	−0.5	0.0	+0.1	96.1	−0.4	96.1	−0.4
	WikiText	23.1	+16.7	73.5	−18.2	87.2	+1.9	74.9	+1.0
	CounterFact	16.6	+2.9	82.8	−2.9	96.5	+0.5	86.6	+0.3
Llama-3.1-8B-Inst	MMLU	67.5	**−1.1**	0.3	−0.2	67.7	**−1.3**	67.7	−0.9
	ARC-Easy	94.1	+0.1	0.0	+0.0	94.1	+0.1	94.0	+0.2
	WikiText	49.8	**+2.7**	45.5	**−3.7**	91.3	−1.2	83.4	−0.5
	CounterFact	20.5	**+2.6**	79.0	**−2.7**	97.5	−0.3	88.1	−0.2

on the Alpaca dataset at batch size 32. The recovery results in Tables 5 and 6 are obtained by running each corresponding method (SFT, SGR, GKD, and KL-regularized self-distillation) for 500 training steps.

Random. The random method prunes layers selected uniformly at random. We report a single random draw (the removed blocks in Table 11) rather than an average over draws, so the random baseline should be read as illustrative.

Reverse (Lu et al., 2024). The reverse method selects the last layers to prune. Lu et al. (2024) report that this is a surprisingly strong method for layer-wise pruning.

Block Influence (Men et al., 2024). Block Influence measures how much a block changes its input hidden state. For block i with input h_i and output h_{i+1},

$$\mathrm{BI}_i = 1 - \mathbb{E}_h \left[\frac{h_i^\top h_{i+1}}{\|h_i\|_2 \, \|h_{i+1}\|_2} \right].$$

Perplexity (Kim et al., 2024). The importance of a block is defined by the increase in perplexity upon its removal.

$$\mathrm{PPL}_i = \mathrm{PPL}\big(\text{model} \setminus \text{block } i\big) - \mathrm{PPL}(\text{model}).$$

Taylor (Ma et al., 2023). The Taylor score is a first-order Taylor estimate of the change in loss $\mathcal{L}$ from pruning W_i,

$$\mathrm{Taylor}_i = \left| \sum_{w \in W_i} \frac{\partial \mathcal{L}}{\partial w} w \right|,$$

Table 11: Blocks removed by each selection rule at 20% pruning of Llama-3.1-8B-Instruct (6 of 32 blocks, 0-indexed).

Method	Removed blocks	Pattern
Random	0, 7, 8, 23, 27, 31	scattered
Reverse	26, 27, 28, 29, 30, 31	last six
BI	23, 24, 25, 26, 27, 28	contiguous late block
PPL	20, 23, 24, 26, 27, 28	late, scattered
Taylor	18, 19, 20, 22, 23, 24	mid-to-late block

C License for Artifacts

All artifacts used in this work are publicly available and were used solely for non-commercial research, consistent with their licenses and intended use. Table 12 lists the models and datasets together with their licenses.

Table 12: Licenses of the models and datasets used in this work.

Artifact	License
Models	
GPT-J-6B	Apache 2.0
Pythia-410M	Apache 2.0
Qwen-2.5-7B-Instruct	Apache 2.0
Llama-2-7B	Llama 2 Community License
Llama-3.1-8B-Instruct	Llama 3.1 Community License
Datasets	
MMLU	MIT
ARC-Easy	CC BY-SA 4.0
WikiText	CC BY-SA 3.0
CounterFact	MIT
GSM8K	MIT
HumanEval	MIT
Alpaca	CC BY-NC 4.0