DepthKV: Layer-Dependent KV Cache Pruning for Long-Context LLM Inference

Anonymous ACL submission

Abstract

Long-context reasoning is a critical capability of large language models enabling applications such as long-document understanding andsummarization, and code generation. However, efficient autoregressive inference is based on the key–value (KV) cache, whose memory footprint grows linearly with sequence length, leading to a major memory bottleneck. To mitigate this overhead, KV cache pruning methods discard cached tokens with low attention scores during inference. Most existing methods apply a uniform pruning ratio across layers, implicitly assuming that all layers contribute equally to the model performance. We show that this strategy is suboptimal as the layers differ significantly in their sensitivity to pruning. We thus propose **DepthKV**, a layer-dependent pruning framework that allocates a fixed global KV budget across layers based on their sensitivity. Across multiple models and tasks, DepthKV consistently outperforms uniform pruning at the same global pruning ratio.

github.com/anonymous-depthkv.git

1 Introduction

Recent advances in large language models (LLMs) have greatly increased context window sizes, ranging from 128K to millions of tokens (Team et al., 2024a; GLM et al., 2024). Larger context windows enable applications such as long-form reasoning, agent-based workflows, and large-scale document retrieval, where relevant information is often sparsely distributed across lengthy inputs.

However, in autoregressive LLMs, each generated token attends to all previous tokens via self-attention. Decoding scales linearly with context length and is repeated at every generation step, while the prefill stage processes the entire input with quadratic complexity. Therefore, as context windows grow, long-context inference becomes prohibitively expensive.

To avoid redundant computation, modern LLMs cache key–value (KV) representations of previous tokens. While this eliminates recomputation during decoding, it introduces a new bottleneck: the KV cache grows with both sequence length, quickly exceeding GPU memory capacity in long-context settings (Wu et al., 2024). As a result, the primary bottleneck shifts from computation to memory. KV pruning methods encounter this chellange by only storing the represntations of important tokens (Shazeer, 2019; Kwon et al., 2023; Ainslie et al., 2023) Most post-training methods assume uniform importance of all layers and prune all layers to the same degree, accordingly. However, previous work (Skean et al., 2025) suggests that intermediate layers may be more critical. Therefore, we conduct experiments in which we prune only single layers and find substantial variations in how much this impacts output quality. Motivated by these findings, we propose **DepthKV**, a framework for layer-dependent KV cache pruning that allocates the memory budget non-uniformly across transformer layers based on their importance for long-context performance, as illustrated in Figure 1. We evaluate DepthKV on long-context tasks including summarization, question answering, and mathematical reasoning, where prefill dominates cost. DepthKV consistently outperforms uniform pruning under the same KV budget.

In summary, our main contributions are as follows:

- We show that transformer layers vary significantly in importance for long-context inference, challenging uniform KV pruning assumptions.
- We find that middle layers are more sensitive to pruning and that the pruning sensitiviy generally correlates with metrices prevously found to indicate layer importance (Skean et al., 2025).

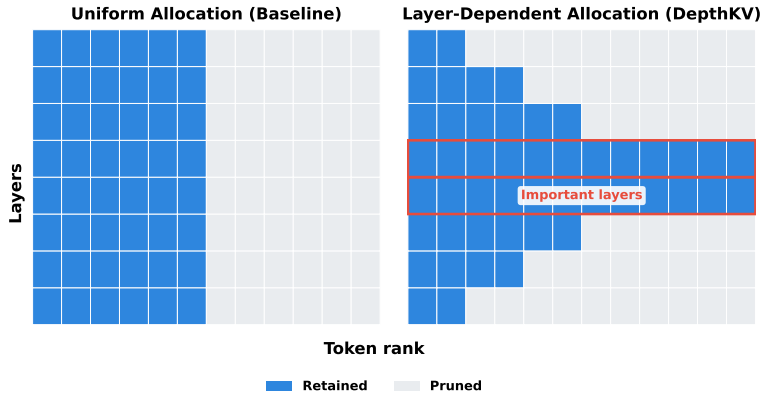

Figure 1: **Uniform vs. layer-dependent KV allocation.** Uniform allocation (left) assigns an equal KV budget across transformer layers. DepthKV (right) reallocates this budget based on sensitivity to pruning, retaining more tokens in critical layers (highlighted) and pruning less important ones more aggressively. Token rank denotes relative importance.

- We propose DepthKV, a layer-dependent KV allocation framework under a fixed memory budget. The framework supports position-based (fixed-region preservation), metric-guided (score-based allocation), and hybrid strategies.
- We demonstrate consistent improvements over uniform pruning across diverse long-context tasks.

2 Related Work

KV cache pruning methods can be categorized along two complementary dimensions: (i) how token importance is estimated, and (ii) whether this importance depends on the current decoding query. The first distinguishes between heuristic and learned methods, while the second separates query-aware from non-query-aware strategies.

Heuristic methods. Heuristic approaches estimate token importance using predefined rules, typically based on positional bias, aggregated attention statistics, or attention profiling across heads or tokens. Representative methods include H_2O (Zhang et al., 2023), StreamingLLM (Xiao et al., 2023), SnapKV (Li et al., 2024), and FastGen (Ge et al., 2023). H_2O retains attention-dominant tokens alongside recent context. StreamingLLM preserves initial attention-sink tokens (i.e., tokens that consistently attract high attention, such as the first token <bos>), together with a sliding window of recent tokens. SnapKV selects prefix tokens based on attention patterns computed from an observation window near the end of the prompt, while FastGen derives head-specific retention policies from attention profiling to determine which tokens to preserve. While these methods are computationally efficient, their reliance on heuristic rules may limit generalization under shifts in the input distribution.

Learned methods. In contrast, learned approaches estimate token importance directly from data rather than predefined rules, enabling them to better capture semantic and task-specific relevance. Representative examples include DuoAttention (Xiao et al., 2024) and SeerAttention (Gao et al., 2024). DuoAttention learns per-head gates to separate retrieval heads with full attention from streaming heads with restricted attention. SeerAttention instead learns input-adaptive block-level sparse attention patterns through a lightweight gating module trained with self-distillation. While generally more flexible, they require an additional training stage and introduce computational overhead.

Query-aware methods. Beyond the above distinction, another key dimension is whether token importance depends on the current decoding query. Most existing KV cache pruning methods are non-query-aware, assigning token importance once when tokens are inserted into the KV cache and reusing it throughout decoding. While computationally efficient, such strategies rely on historical information or current states to decide which tokens to discard, making these decisions effectively irreversible. As a result, tokens that appear unimportant at early stages may later become critical for future decoding steps, leading to the loss of relevant information. A separate line of work focuses on query-aware strategies. Query-aware methods

dynamically estimate importance at each decoding step, enabling adaptive retrieval but at the cost of additional computation. Representative approaches include Quest (Tang et al., 2024), RetrievalAttention (Liu et al., 2024), and MorphKV (Ghadia et al., 2025). Quest partitions the KV cache into fixed-size pages (i.e., groups of tokens) and uses the current query together with page-level key summaries to estimate the most relevant pages. RetrievalAttention selects the most relevant KV entries for the current query using vector search over indexed keys. MorphKV iteratively updates a fixed-size KV cache based on recent attention patterns.

Positioning of DepthKV. Within the above taxonomy, DepthKV can be viewed as a non-query-aware, heuristic post-training method, but differs from prior work by focusing on layer-wise sensitivity instead of token-level importance. DepthKV allocates the KV cache budget across layers based on this sensitivity, achieving a superior memory–performance trade-off and consistently outperforming uniform pruning under the same global pruning ratio.

3 KV cache pruning

In this section, we formalize KV cache pruning under a global budget and review attention-based pruning strategies.

3.1 Problem Formulation

Consider a decoder-only transformer with L layers and hidden dimension d. For input length N, each layer produces key and value matrices , $K^{(l)}$ and $V^{(l)} \in \mathbb{R}^{N\times d}$, respectively, requiring $\mathcal{O}(LNd)$ memory. Let $S^{(l)} \subseteq \{1,\dots,N\}$ with $|S^{(l)}| = B^{(l)}$ denote the retained tokens at layer l. The KV memory footprint is proportional to $\sum_{l=1}^{L} B^{(l)} d$. We seek $S^{(l)}$ and $B^{(l)}$ that keep the global budget $\sum_{l=1}^{L} B^{(l)} = B_{\text{total}}$ constant while affecting the performance as little as possible.

3.2 Attention-Based Token Importance Estimation

Previous methods retain the same number of tokens in each layer, i.e. they set $B^{(l)} = \frac{B_{total}}{L}, \forall l = 1,\dots,L$, and propose to estimate token importance from attention weights during prefill. Let $Q_i^{(l)}$, $K_j^{(l)}$, and $V_j^{(l)}$ denote query, key, and value vetors of the i-th token at layer l. The attention score and weights are then given by

$$a_{i,j}^{(l)} = \frac{\langle Q_i^{(l)}, K_j^{(l)}\rangle}{\sqrt{d_k}} \text{ and } \alpha_{i,j}^{(l)} = \frac{\exp(a_{i,j}^{(l)})}{\sum_{t\le i}\exp(a_{i,t}^{(l)})}.$$

Following the Heavy-Hitter Oracle(Zhang et al., 2023) (to which we refer to as **H_2O w/o V**), token importance is defined as the cumulative attention from later tokens. That is, the importance of token j at layer l is given by $s_j^{(l)} = \sum_{i=j+1}^{N} \alpha_{i,j}^{(l)}$. In the multi-head setting, attention is aggregated across heads before computing scores. Under this formulation, the last token in each chunk receives zero importance; however, chunked prefill allows tokens to be revisited in later chunks, mitigating this effect. We do not enforce a local window. The $B^{(l)}$ tokens with the highest scores are then retained. (Guo et al., 2024) modify the approach by also take the norm of the value vectore into account: $s_j^{(l)} = \|V_j^{(l)}\|_p \sum_{i=j+1}^{N} \alpha_{i,j}^{(l)}$, where $\|\cdot\|_p$ is a vector norm. We refer to the resulting pruning nudget allocation stategy as **H_2O w/ V** and consider $p = 1$ and $p = 2$, using $p = 1$ (i.e. the ℓ_1 norm) by default unless otherwise stated.

4 Pre-study: Layer-wise Sensitivity to KV Cache Pruning

The analysis of (Skean et al., 2025) found that the intermediate layers of the transformer play a more important role than the early or late layers. Therefore, the question arises whether this also translates in a non-uniform sensitivity to KV cache pruning. To answer this question, we performed experiments on different models and datasets (described in Section 6) where KV prefill-stage cache pruning is applied one layer at a time, keeping the other layers unchanged. That is, for each layer l, we apply H_2O w/o V pruning and record different performance measures (also reported in Section 6), producing a layer-wise sensitivity profiles. The sensitivity profiles for the ROUGE-1 measure on several summarization datasets are shown in Figure 2, where the ROUGE-1 values resulting from the pruning of different layers differ greatly, with sharp performance drops observed for specific layers. The layers inducing the most heavy performance drops vary heavily between models and to a lesser extent between datasets. To test for statistical significance of the layer-wise performance differences we performed a permutation test. The null hypothesis of uniform pruning-sensitivity is rejected with a p-

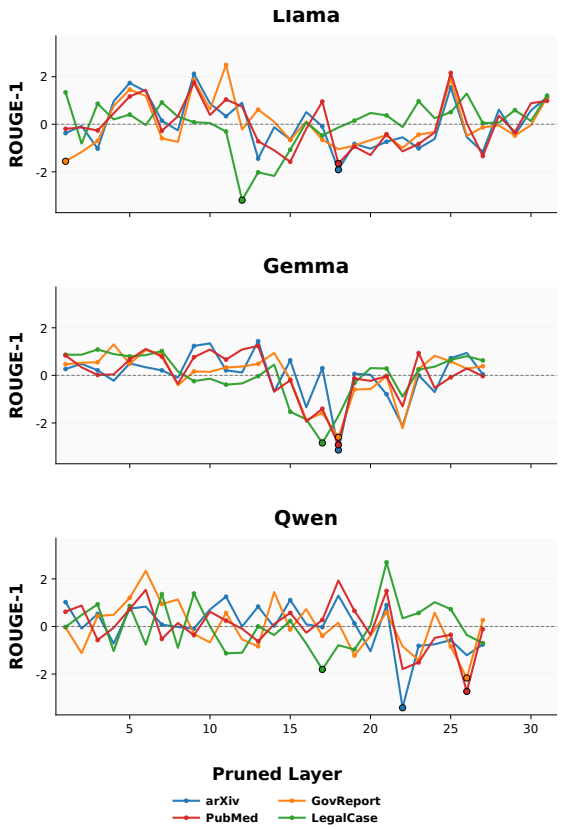

Figure 2: **Single-layer KV cache pruning.** Layer-wise ROUGE-1 under KV cache pruning of individual layers, standardized within each model–dataset pair (z-score; mean = 0, standard deviation = 1). Markers indicate the layer with the largest performance drop for each dataset.

value < 0.05 (see Appendix F) in all experiments.

We also measued the YapScore for the different models when each layer is independently pruned during prefill. The results show that pruning certain layers suppresses generation, producing shorter or incomplete outputs that degrade summary quality. Over different datasets and layers causing larger YapScore drops align with those causing the largest performance drop. Table 1 quantifies this via the layer-wise correlation between YapScore and ROUGE-1. Qualitative examples are provided in Appendix E. Overall, these results show that layers differ in sensitivity to KV cache pruning, motivating layer-dependent KV allocation.

4.1 Representation Metrics for Layer Importance

To explain the observed layer-wise variation, we analyze representation properties using metrics from Skean et al. (2025), capturing information, geometry, and invariance. We consider all six of their suggested metrics: spectral entropy, effective rank, curvature, DiME, LiDAR, and InfoNCE. We detail InfoNCE below due to its central role in our correlation analysis; other metrics are described in Appendix G. InfoNCE measures the representation invariance to perturbations of single tokens, normalized by the difference of representations of different tokens For each input, the batch representation matrix $Z \in \mathbb{R}^{T\times d}$ is extracted from the post-attention stage, where T is the sequence length, d the hidden dimension, and the row z_i is the representation vector of the i-th token. All representations are ℓ_2-normalized. Then pairs $(\bar{z}_i^{(o)}, \bar{z}_i^{(a)})$ of normalized representations of original and perturbed input (10% of the words dropped uniformly at random) are formed and the remaining representations of the batch $\bar{z}_j^{(o)} : j \neq i$ are collected. Then the **InfoNCE** loss for sample i is defined as

$$\mathcal{L}_{\text{InfoNCE}}^{(i)} = -\log \frac{\exp\Big(\text{sim}(\bar{z}_i^{(o)}, \bar{z}_i^{(a)})/\tau\Big)}{\sum_j \exp\Big(\text{sim}(\bar{z}_i^{(o)}, \bar{z}_j^{(o)})/\tau\Big)} \quad ,$$

where $\text{sim}(u, v) = u^\top v$ denotes the cosine similarity and τ is a temperature parameter.

We evaluate InfoNCE and related metrics (Skean et al., 2025) as proxies for the layer importance by measuring their correlation with performance degradation under KV cache pruning. Metrics are computed at four transformer stages (pre-attention, post-attention, post-residual, post-MLP) and compared to layer-wise performance drops. The full results are shown in Appendix H. Overall, representation metrics often show statistically significant correlations with degradation, indicating their relevance for layer importance. Among them, InfoNCE is the most consistent, achieving the highest number of significant correlations, especially at the post-attention stage. As shown in Figure 3, InfoNCE exhibits a consistent negative correlation

	GovReport		LegalCase	
Model	r	p	r	p
GEM7	0.5202	2.49×10^{-4}	0.9902	2.89×10^{-39}
LAM8	0.9150	3.78×10^{-22}	0.7269	4.86×10^{-10}
QWEN7	0.7295	8.90×10^{-9}	0.7920	5.57×10^{-11}

Table 1: **ROUGE-1–YapScore Correlation.** Pearson correlation coefficients (r) and corresponding p-values between ROUGE-1 and YapScore, computed across layers under KV cache pruning.

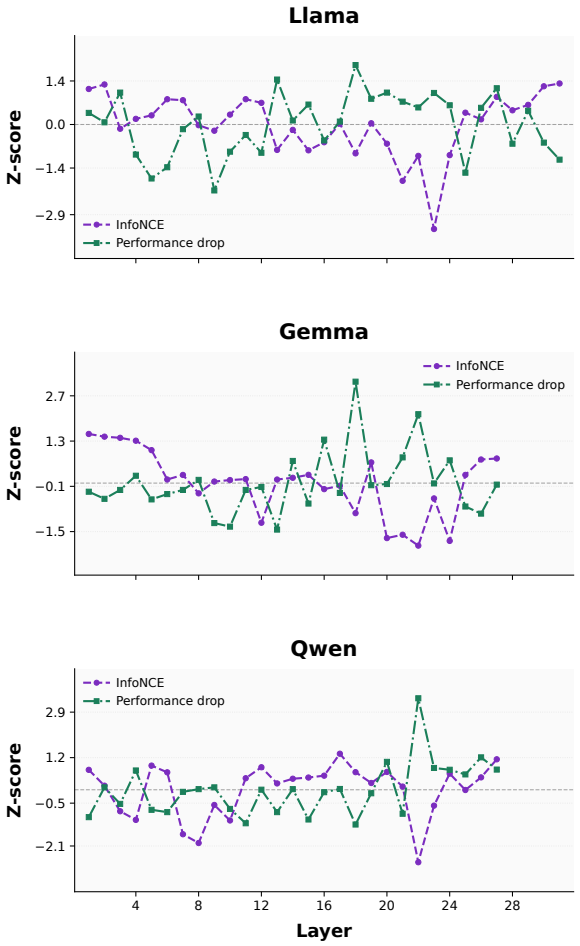

Figure 3: **Layer Importance vs. InfoNCE.** Standardized InfoNCE (post-attention) and ROUGE-1 performance drop across layers under KV cache pruning on the arXiv dataset.

with performance degradation: layers with lower InfoNCE incur larger drops when pruned. This suggests that more perturbation-robust representations are more pruning-sensitive.

5 DepthKV: Layer-Dependent KV Allocation

Since our pre-study shows that sensitivity to KV cache pruning varies across layers and can be predicted from representation properties, we propose **DepthKV**, which allocates the global KV budget B_{total} by layer importance. We do not prune the first layer to preserve the initial representations (Devoto et al., 2024). We consider three complementary strategies:

Middle-Layer Protection (MLP). Motivated by the work of (Skean et al., 2025) that highlights the importance of intermediate layers and supported by our analysis, we preserve layers around the network midpoint—specifically $\lfloor L/2 \rfloor$ and $\lfloor L/2 \rfloor + 1$—while uniformly pruning the others, keeping the representations of $\frac{B_{total}}{L-3}$ tokens in each layer.

Metric-Guided Allocation (MGA). Since we found that InfoNCE correlates with the layer importance, we compute for each layer l, the average InfoNCE loss $\mathcal{L}^{(l)}$ over all tokens and define the layer importance score as $s^{(l)} = -\mathcal{L}^{(l)}$. We then normalize the scores across pruned layers $\hat{s}^{(l)} = \frac{s^{(l)}}{\sum_{j=2}^{L} s^{(j)}}$, Each layer then keeps the $B^{(l)} = \lfloor B_{total} \cdot \hat{s}^{(l)} \rfloor$ tokens with the highest token importance $s_j^{(l)}$ calculated as in the H_2O w/o V method. However, each layer cannot retain a higher number of tokens than that given by the secenque length $B_{max} = N$. Moreover, to ensure that no layer is too excessively pruned, we define a minimum number B_{min} of tokens that need to be retained in each layer. To ensure that $B^{(l)}$ falls into the interval $[B_{min}, B_{max}]$, we apply clipping and redistribute the remaining budget proportionally to the layer importance of that layers whose budget is not already at the minimum or maximum . This procedure is repeated until the total budget B_{total} is distributed.

Middle-Layer Metric Allocation (MLMA). In this approach, we combine structural and metric-based allocation by preserving a subset of middle layers while distributing the remaining KV budget across the other layers using InfoNCE-based importance scores. Specifically, MLMA-2L preserves two middle layers.

6 Experimens

We now describe the experimental setting.

6.1 Datasets

We consider four long-document summarization benchmarks spanning scientific, biomedical, legal, and government domains (arXiv and PubMed (Cohan et al., 2018), GovReport (Huang et al., 2021), and LegalCase (Shukla et al., 2022)), along with two document-grounded QA benchmarks (Qasper (Dasigi et al., 2021) and HotpotQA (Yang et al., 2018)), and a synthetic mathematical reasoning benchmark (GSM-∞). Table 2 summarizes key statistics for all datasets, including input length (mean $\pm$ std), task type, and the range of input lengths. For summarization, we evaluate on 1,000 randomly sampled documents with input lengths

Dataset	Task	Avg ± Std	Min	Max
arXiv	Sum.	4926.7 ± 1064.2	3053	7396
GovReport	Sum.	5797.2 ± 1177.0	3841	8042
LegalCase	Sum.	5384.7 ± 1131.5	3726	7969
PubMed	Sum.	5092.2 ± 1060.6	3400	7809
HotpotQA	QA	934.7 ± 268.5	372	1674
Qasper	QA	2193.0 ± 601.3	774	2960
GSM-∞	Reason.	2620.9 ± 816.2	825	4240

Table 2: **Input length statistics across datasets.** All lengths are measured in words. Mean ± standard deviation, minimum, and maximum are shown. Tasks: Sum. (summarization), QA (question answering), Reason. (mathematical reasoning). Rows are color-coded by task category.

ranging from 5K to 10K tokens. For QA and reasoning tasks, we restrict inputs to at most 4K tokens by selecting only examples below this length threshold. We use 1,000 samples for HotpotQA and 100 and 500 samples for Qasper and GSM-∞, respectively, due to the limited size of the filtered datasets.

6.2 Models

We evaluate our method on three models from widely used open-weight LLM families: Gemma, LLaMA, and Qwen, which represent diverse architectures and training paradigms. Due to computational constraints, we evaluate one representative model per family: *Gemma-7B-it* (Team et al., 2024b), *Llama-3.1-8B-Instruct* (Meta AI, 2024), and *Qwen2.5-7B-Instruct* (Hui et al., 2024).

6.3 Evaluation Metrics

We evaluate summarization quality we report ROUGE-1, ROUGE-2, and ROUGE-L (Lin, 2004); SBERT-based semantic similarity (Reimers and Gurevych, 2019); and YapScore (Borisov et al., 2026). ROUGE measures the lexical overlap between generated and reference summaries, while SBERT captures semantic similarity through cosine similarity between sentence embeddings. In addition, YapScore measures output length relative to a fixed baseline, allowing us to characterize pruning-induced suppression of generated content and examine its association with downstream performance degradation. Full details are provided in Appendix D. For QA and reasoning datasets, we report exact match (EM) accuracy. On HotpotQA and Qasper, we additionally compute token-overlap precision, recall, and F1 to capture partial correctnessPrior work suggests that token-overlap metrics remain useful for generative QA, with recall being particularly well aligned with human judgments (Adlakha et al., 2024).

6.4 Implementation Details

All pruning-based methods are evaluated under a fixed global KV cache reduction ratio of 60%, translating into a total budget of $B_{total} = 40\%$.

We use deterministic greedy decoding (`do_sample=False`), selecting the highest-probability token at each step to ensure reproducibility. The maximum generation length is set to 500 tokens, with early termination upon generation of the end-of-sequence token. All inputs are processed using chunked prefill with a fixed chunk size of 1024 tokens for long-context evaluation. After each chunk, token importance scores are updated and used to prune the KV cache. The KV cache remains fixed during decoding.

All experiments were conducted on a single compute node equipped with 8× NVIDIA H200 GPUs.

7 Results

We evaluate DepthKV first on summarization tasks, then on QA and reasoning tasks, and finally assess its output quality using an LLM-as-a-judge framework.

7.1 DepthKV Performance on Summarization

We evaluate DepthKV on long-document summarization in comparison to uniform pruning baselines. Table 3 shows that MGA consistently performs well across both datasets, achieving the best overall performance and demonstrating that representation-based signals provide a reliable estimate of layer importance. For example, on arXiv, MGA improves ROUGE-1 from 26.75 to 29.75 and SBERT from 55.09 to 61.98, with similar gains observed on GovReport. Structure-based allocation strategies show less consistent behavior. While all approaches improve performance over uniform pruning on arXiv, their impact on GovReport is less consistent, with certain strategies even degrading performance. This suggests that the assumption that middle layers are most pruning sensitive does not generalize across datasets. We also observe that value-aware variants (w/ V) yield only marginal gains, whereas the primary improvements stem from how the KV budget is distributed across layers. This indicates that layer-wise allocation is

	arXiv			
	R1	R2	RL	SB
FullKV	39.09	13.84	22.57	79.42
w/o V	26.75	5.50	17.24	55.09
w/ V (ℓ_1)	26.84	5.58	17.21	53.90
w/ V (ℓ_2)	26.63	5.42	17.06	54.00
MGA	29.75	6.92	18.59	61.98
MLMA-2L	29.54	6.55	18.52	61.19
MLP	28.47	6.01	17.99	57.96

	GovReport			
	R1	R2	RL	SB
FullKV	37.61	13.57	19.22	87.25
w/o V	26.76	5.98	15.68	62.05
w/ V (ℓ_1)	27.03	5.89	15.74	61.53
w/ V (ℓ_2)	26.75	5.98	15.64	61.65
MGA	28.43	7.05	16.36	70.24
MLMA-2L	24.13	5.96	14.06	65.90
MLP	23.24	5.41	13.90	61.82

Table 3: **Summarization results on Gemma.** FullKV denotes the unpruned KV-cache; all other methods follow Sec. 3. R-1, R-2, and R-L are ROUGE scores, and SB denotes Sentence-BERT similarity. All values are reported in %; best results are highlighted in green.

more important than the specific token importance estimator.

Overall, these results demonstrate that DepthKV effectively improves summarization quality, validating the use of representation-based signals to guide layer-wise KV allocation.

7.2 Generalization to QA and Reasoning

To assess robustness beyond summarization, we evaluate DepthKV on document-grounded question answering and mathematical reasoning.

As shown in Table 4, DepthKV consistently outperforms uniform pruning, while the best allocation strategy depends on the model and dataset. On HotpotQA, MLP achieves the highest accuracy for Gemma, while MGA performs best for LLaMA. On Qasper, MGA achieves the strongest overall results for Gemma, whereas for LLaMA, MGA and MLP obtain the same accuracy, with MGA achieving higher recall and F1 scores.

Together, these results indicate improved preservation of relevant information for multi-step reasoning and document-grounded QA, as illustrated by qualitative examples in Appendix E.

Beyond QA, DepthKV also improves performance on mathematical reasoning tasks, with all variants outperforming uniform pruning on GSM-∞ (Figure 4).

Overall, these results show that the benefits of DepthKV extend beyond summarization to both QA and reasoning tasks, demonstrating robustness across diverse settings.

7.3 LLM-as-a-Judge Evaluation

To complement automatic metrics, we further evaluate answer quality using an LLM-as-a-judge

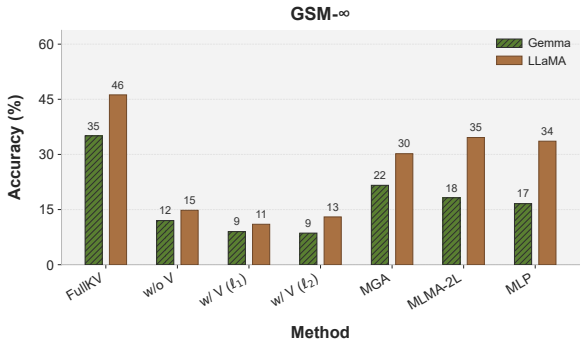

Figure 4: **GSM-∞ accuracy.** Performance on the GSM-∞ benchmark across different KV cache pruning settings.

framework. As shown in Table 5, MGA consistently achieves the highest scores across all evaluation dimensions, while all DepthKV variants generally outperform uniform pruning baselines. These findings are consistent with the automatic metrics on HotpotQA, where MGA and MLP are the strongest-performing variants. Overall, the results indicate that the best-performing DepthKV variants improve both quantitative performance and answer quality.

8 Conclusion

We investigate KV cache pruning for long-context LLM inference and show that assuming uniform layer importance is suboptimal. Through layer-wise ablation, we demonstrate that transformer layers vary significantly in their sensitivity to KV pruning and identify content amplification layers that are critical for preserving information.

We further find that representation-level metrics provide effective signals of layer importance. Motivated by these insights, we introduce **DepthKV**,

Method	HotpotQA								Qasper							
	GAM7				LAM8				GAM7				LAM8			
FullKV	57	60	58	55	79	77	78	72	83	35	49	50	84	61	71	65
w/o V	14	16	14	12	59	54	55	47	12	6	8	6	64	72	68	54
w/ V (ℓ_1)	15	17	15	13	56	52	53	46	12	6	8	6	66	76	71	58
w/ V (ℓ_2)	14	15	14	12	57	52	54	46	9	4	6	5	66	76	71	58
MGA	20	21	20	18	77	73	74	67	57	34	43	38	66	82	73	60
MLMA-2L	10	12	11	10	56	52	53	45	40	15	22	27	67	81	73	60
MLP	25	28	26	23	76	73	74	66	42	15	22	28	69	73	71	60

Table 4: **Comparison of methods on HotpotQA and Qasper.** Each cell reports Precision, Recall, F1, and Accuracy (in %).

Method	GEM7			LAM8		
	CR	CP	CN	CR	CP	CN
FullKV	4.16	4.37	4.80	3.39	3.74	4.80
w/o V	1.63	1.66	1.71	3.59	3.60	3.76
w/ V (ℓ_1)	1.64	1.67	1.79	3.54	3.56	3.76
w/ V (ℓ_2)	1.65	1.69	1.76	3.55	3.56	3.78
MGA	2.67	2.63	2.61	4.40	4.38	4.55
MLMA-2L	1.70	1.71	1.89	3.73	3.76	4.03
MLP	2.44	2.44	2.56	4.36	4.36	4.53

Table 5: **LLM-as-a-Judge evaluation on HotpotQA.** Scores for correctness (CR), completeness (CP), and conciseness (CN) on a 1–5 scale; evaluation criteria follow Appendix C.

a layer-dependent KV pruning framework that reallocates a fixed global KV budget across layers via multiple allocation strategies.

Across summarization, QA, and reasoning tasks, DepthKV consistently outperforms uniform pruning under the same memory budget, improving both automatic metrics and LLM-as-a-judge evaluations. Overall, our results show that accounting for the heterogeneous roles of transformer layers leads to more efficient KV cache usage and offers a simple, general approach to achieving this.

9 Limitations

While the proposed method demonstrates promising performance, several limitations should be considered.

First, DepthKV operates in a non-query-aware setting, where token importance is estimated during the prefill stage without conditioning on the decoding query. However, token relevance may change during autoregressive decoding. As a result, DepthKV may overlook tokens that become important only for specific queries, limiting performance in retrieval-intensive or fine-grained reasoning tasks. Incorporating query-aware token selection into the layer-wise KV cache allocation could address this limitation.

Second, DepthKV relies on heavy-hitter-based token importance estimation by aggregating attention scores across heads, which may obscure head-specific behaviors. Prior work suggests that attention heads often serve specialized roles and contribute unequally (Ge et al., 2023). Although DepthKV allocates cache budgets across layers, it does not capture intra-layer variability across heads. Extending the framework to jointly model layer-wise and head-wise importance is a natural direction for improvement.

References

Vaibhav Adlakha, Parishad BehnamGhader, Xing Han Lu, Nicholas Meade, and Siva Reddy. 2024. Evaluating correctness and faithfulness of instruction-following models for question answering. *Transactions of the Association for Computational Linguistics*, 12:681–699.

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

Vadim Borisov, Michael Gröger, Mina Mikhael, and Richard H Schreiber. 2026. Do chatbot llms talk too much? the yapbench benchmark. *arXiv preprint arXiv:2601.00624*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model

for abstractive summarization of long documents. *arXiv preprint arXiv:1804.05685*.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers.

Alessio Devoto, Yu Zhao, Simone Scardapane, and Pasquale Minervini. 2024. A simple and effective l_2 norm-based strategy for kv cache compression. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18476–18499.

Yizhao Gao, Zhichen Zeng, Dayou Du, Shijie Cao, Peiyuan Zhou, Jiaxing Qi, Junjie Lai, Hayden Kwok-Hay So, Ting Cao, Fan Yang, and 1 others. 2024. Seerattention: Learning intrinsic sparse attention in your llms. *arXiv preprint arXiv:2410.13276*.

Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann LeCun. 2023. Rankme: Assessing the downstream performance of pretrained self-supervised representations by their rank. In *Proceedings of the International Conference on Machine Learning*, pages 10929–10974. PMLR.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. 2023. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*.

Ravi Ghadia, Avinash Kumar, Gaurav Jain, Prashant Nair, and Poulami Das. 2025. Dialogue without limits: Constant-sized kv caches for extended responses in llms. *arXiv preprint arXiv:2503.00979*.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, and 1 others. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Zhiyu Guo, Hidetaka Kamigaito, and Taro Watanabe. 2024. Attention score is not all you need for token importance indicator in kv cache reduction: Value also matters. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21158–21166.

Eghbal Hosseini and Evelina Fedorenko. 2023. Large language models implicitly learn to straighten neural sentence trajectories to construct a predictive representation of natural language. *Advances in Neural Information Processing Systems*, 36:43918–43930.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. *arXiv preprint arXiv:2104.02112*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, and 1 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2409.12186*.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4334–4353.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. 2024. Snapkv: Llm knows what you are looking for before generation. *Advances in Neural Information Processing Systems*, 37:22947–22970.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81.

Di Liu, Meng Chen, Baotong Lu, Huiqiang Jiang, Zhenhua Han, Qianxi Zhang, Qi Chen, Chengruidong Zhang, Bailu Ding, Kai Zhang, and 1 others. 2024. Retrievalattention: Accelerating long-context llm inference via vector retrieval. *arXiv preprint arXiv:2409.10516*.

Meta AI. 2024. Meta llama 3.1. https://ai.meta.com/blog/meta-llama-3. Accessed 2024.

Belinda Phipson and Gordon K Smyth. 2016. Permutation p-values should never be zero: calculating exact p-values when permutations are randomly drawn. *arXiv preprint arXiv:1603.05766*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Keita Saito, Akifumi Wachi, Koki Wataoka, and Youhei Akimoto. 2023. Verbosity bias in preference labeling by large language models. *arXiv preprint arXiv:2310.10076*.

Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*.

Abhay Shukla, Paheli Bhattacharya, Soham Poddar, Rajdeep Mukherjee, Kripabandhu Ghosh, Pawan Goyal, and Saptarshi Ghosh. 2022. Legal case document summarization: Extractive and abstractive methods and their evaluation. *arXiv preprint arXiv:2210.07544*.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. 2025. Layer by layer: Uncovering hidden representations in language models. *arXiv preprint arXiv:2502.02013*.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. 2024. Quest: Query-aware sparsity for efficient long-context llm inference. *arXiv preprint arXiv:2406.10774*.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, and 1 others. 2024a. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, and 1 others. 2024b. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Vimal Thilak, Chen Huang, Omid Saremi, Laurent Dinh, Hanlin Goh, Preetum Nakkiran, Joshua M. Susskind, and Etai Littwin. 2023. Lidar: Sensing linear probing performance in joint embedding self-supervised learning architectures. *arXiv preprint arXiv:2312.04000*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and 1 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2024. Retrieval head mechanistically explains long-context factuality. *arXiv preprint arXiv:2404.15574*.

Guangxuan Xiao, Jiaming Tang, Jingwei Zuo, Junxian Guo, Shang Yang, Haotian Tang, Yao Fu, and Song Han. 2024. Duoattention: Efficient long-context llm inference with retrieval and streaming heads. *arXiv preprint arXiv:2410.10819*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, and 1 others. 2023. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710.

A Reproducibility and Licenses

A.1 Dataset Licenses and Usage

The datasets utilized in this study are obtained from publicly accessible repositories and are used in compliance with their respective licensing terms, as summarized below.

1. **arXiv, PubMed, GovReport:** Accessed via ccdv on Hugging Face, these datasets are released under the Apache-2.0 License.

2. **LegalCase:** Provided by the authors at Law-AI (GitHub), this dataset is constructed from publicly available Indian and U.K. Supreme Court decisions and adheres to the terms of the original sources.

3. **Qasper:** Available via AllenAI on Hugging Face, the dataset is distributed under the CC-BY-4.0 License.

4. **HotpotQA:** Obtained from HotpotQA on Hugging Face, the dataset is released under the CC-BY-SA-4.0 License.

5. **GSM-∞:** Released by the authors at Infini-AI-Lab (GitHub), this synthetic dataset is generated programmatically and is used in accordance with the terms specified in the repository.

A.2 Model Licenses

We employ open-weight language models accessed via the Hugging Face Transformers library (Wolf et al., 2020). Specifically, the models used are:

- `google/gemma-7b-it`, released under the Gemma Terms of Use;
- `meta-llama/Llama-3.1-8B-Instruct`, provided under the LLaMA 3.1 Community License; and
- `Qwen/Qwen2.5-7B-Instruct`, distributed under Apache License 2.0.

All models are used in accordance with their respective licensing terms.

Our implementation builds upon the Hugging Face Transformers framework to incorporate the proposed pruning methods. To facilitate reproducibility, all source code, model configurations, and experiment scripts are made publicly available in the GitHub repository referenced in the abstract.

B Prompt Templates

To ensure reproducibility, we specify the exact prompts used in our experiments. These prompts were applied consistently during both inference and evaluation, with no modifications, and no additional system prompts unless explicitly stated. Placeholders (e.g., `DOCUMENT`, `CONTEXT`, `QUESTION`) were instantiated with the corresponding inputs from each dataset. The following subsections present dataset-specific prompt templates across summarization, QA, and reasoning tasks.

B.1 Generation Prompts

The following prompts were used to generate model outputs during inference.

B.1.1 Summarization Prompt Templates

For the long-document summarization datasets (arXiv and GovReport), we employ a generic summarization instruction, as they span diverse domains and exhibit varying document structures. In contrast, the PubMed dataset consists exclusively of biomedical research articles with a standardized structure; accordingly, the prompt explicitly refers to a scientific article. Similarly, for the LegalCase dataset, the prompt specifies that the input is a legal judgment to reflect its domain.

arXiv and GovReport

```
Summarize the following document:

{DOCUMENT}

Summary:
```

PubMed

```
Summarize the following scientific article:

{DOCUMENT}

Summary:
```

LegalCase

```
Summarize the following legal judgment:

{DOCUMENT}

Summary:
```

B.1.2 QA & Reasoning Prompts

For QA and reasoning tasks, prompt templates impose strict answer-format constraints intended to encourage consistent and automatically evaluable outputs. In HotpotQA, the model is instructed to provide only a short answer without explanation or additional text, while in Qasper it is instructed to respond with a single word (yes or no). Similarly, for GSM-∞, the prompt directs the model to return only the final numeric answer, without intermediate reasoning or repeated text. These constraints help reduce output variability and improve evaluation reliability.

HotpotQA

```
Read the following context and answer
the question using ONLY the short answer.
Do not include explanations.  Your
answer MUST start with 'answer:'  and
must not include additional words.

Context:  {CONTEXT}
Question:  {QUESTION}

Answer:
```

Qasper

```
You are answering a strict binary
scientific question.  You must respond
with exactly ONE WORD. Allowed answers:
yes or no.
Do NOT write explanations.
Do NOT write sentences.
Do NOT write 'sometimes', 'partially',
or any other word.
If uncertain, choose the most likely
between yes or no.

Title:  {TITLE}
Abstract:  {ABSTRACT}
Full Text:  {FULL TEXT}
Question:  {QUESTION}

Answer:
```

GSM-∞

```
Solve the following math problem
carefully.
Return only the final numeric answer.
Do not include explanation, reasoning,
or repeated text.

Problem:  {PROBLEM}
Question:  {QUESTION}

Final answer:
```

B.2 Evaluation Prompts

The following prompt template is used for LLM-as-a-judge evaluation within the framework described in Appendix C. It enforces a standardized output format, requiring a brief justification followed by a discrete score.

```
You are an expert evaluator of question
answering systems.
Your task is to evaluate the {METRIC
NAME} of the model answer by comparing
it to the reference answer.
You must:

  - Compare the model answer with the
    reference answer.
  - Consider semantic equivalence, not
    exact wording.
  - Follow the rubric strictly.
  - Provide concise reasoning (max 3
    sentences).

### Question
{QUESTION}

### Reference Answer
{REFERENCE}

### Model Answer
{PREDICTION}

### Evaluation Metric
{METRIC NAME}

### Scoring Rubric
{RUBRIC}

You MUST output EXACTLY in this format:

Reasoning:  <max 3 sentences>

Final Score:  <one digit 1-5>

Nothing else.
```

C Evaluation with Prometheus

We evaluate model outputs using an LLM-as-a-judge framework, where a separate language model assesses the quality of generated answers. For this purpose, we use Prometheus (Kim et al., 2024), an open-source model trained to evaluate the outputs of other models according to predefined rubrics. Specifically, we employ the `prometheus-eval/prometheus-8x7b-v2.0` model, which demonstrated more stable and consistent performance than smaller non-MoE variants (e.g., `prometheus-eval/prometheus-7b-v2.0`) in preliminary experiments. All evaluations are performed using deterministic decoding to ensure reproducibility.

This framework is applied to the HotpotQA dataset, where answers are short and reference answers are available for direct comparison. For each instance, the evaluator receives the question, reference answer, and model prediction, then assigns

scores according to the rubric. The corresponding prompt template is provided in Appendix B.2, and the rubric uses three dimensions—Correctness, Completeness, and Conciseness—each scored independently on a 1–5 scale, as defined below.

Correctness (CR) – Accuracy of the answer relative to the reference answer.

Evaluate whether the model answer matches the reference answer.

Score 1: Completely incorrect or unrelated answer.
Score 2: Mostly incorrect with little overlap.
Score 3: Partially correct but missing key elements.
Score 4: Mostly correct with minor deviations.
Score 5: Fully correct and equivalent to the reference answer.

Completeness (CP) – Coverage of required information.

Evaluate whether the answer fully covers the information contained in the reference answer.

Score 1: Missing all essential information.
Score 2: Missing most key information.
Score 3: Contains some key information but incomplete.
Score 4: Covers most required information.
Score 5: Fully covers all information in the reference answer.

Conciseness (CN) – Directness and brevity.

Evaluate whether the answer is short and directly addresses the question without unnecessary text.

Score 1: Extremely verbose or irrelevant.
Score 2: Some unnecessary explanation.
Score 3: Acceptable length but slightly verbose.
Score 4: Concise with minimal extra content.
Score 5: Very concise and directly answers the question.

D Verbosity Analysis Using YapScore

We quantify output verbosity using YapScore (Borisov et al., 2026), which measures the number of generated tokens exceeding a predefined baseline length. This is particularly relevant in evaluation settings, where LLM-based judges have been shown to prefer longer responses even when shorter answers are equally informative (Saito et al., 2023). Formally, for an output of length L and a baseline B, the YapScore is defined as:

$$\text{YapScore} = \max(0, L - B).$$

E Qualitative Examples

We provide qualitative examples to complement the quantitative results and offer additional insight into model behavior under different KV cache pruning strategies. Specifically, we present two types of comparisons: (i) method-level comparisons between H_2O (w/o v) and MGA, highlighting differences in output accuracy and informativeness, and (ii) layer-wise comparisons illustrating how pruning different transformer layers affects generation behavior, particularly in terms of content preservation and verbosity.

Method Comparison. We provide qualitative comparisons between H_2O (w/o v) and MGA, demonstrating that MGA more effectively preserves key information and yields more accurate outputs in both QA and summarization tasks.

Example 1

Task: QA Model: GEM7 Dataset: arXiv

Prompt

```
Which American politician did
Donahue replace?
```

Reference

Reference Answer

```
kelli ward
```

MGA Output

```
sure, here is the answer: donahue
replaced kelli ward in the
arizona state senate.
```

H_2O (w/o v) Output

```
member of the vermont house of
representatives since 2002.
```

Example 2

Task: Summarization Model: GEM7 Dataset: arXiv

Input Document

```
Summarize the following document:

Document:

One of the oldest problems in
algebra is the equation
solvability problem over
algebraic structures. We study
its computational complexity over
finite groups and rings. It asks
whether two group expressions can
attain the same value for some
substitution, while the
equivalence problem asks whether
two expressions define the same
function. Prior work gives
polynomial-time and hardness
results for some cases, but many
remain open. We show that
equation solvability over
semipattern groups is solvable in
polynomial time.

[Document truncated for brevity]
```

MGA Output

```
Sure, here is a summary of the
document:\n\nThis document
investigates the complexity of
the equation solvability over
finite groups. The main results
are as follows:\n\n* The equation
solvability problem over a finite
group G is in the complexity
class NP.\n* The problem is
complete for the class NP if G is
non-solvable.\n* The problem is
also complete for the class NP if
G is a finite group.\n\nThe
document also includes some open
problems and directions for
future research.
```

H_2O (w/o v) Output

```
The paper is well-structured and
easy to follow, but it is a bit
long and complex. The main result
is that the equation of the
form\n\nA_n = \\sum_{i=1}^n a_i
x_i is solvable in polynomial
time if and only if the pattern
of the polynomial is regular.
This result is obtained by
applying the techniques of
algebraic geometry and the theory
of pattern complexity.\n\nThe
paper is divided into several
sections, each one dealing with a
particular aspect of the problem.
The first section introduces the
problem and the main result. The
second section provides a
reduction to the case of a single
variable. The third section
provides a reduction to the case
of a single variable. The fourth
section provides a reduction to
the case of a single variable.
The fifth section provides a
reduction to the case of a single
variable. [...]
```

Layer Pruning and Content Amplification. To better understand how pruning affects generation behavior, we present qualitative examples comparing outputs when different layers are pruned. These examples illustrate that pruning certain layers can lead to substantial reductions in output length and, more importantly, the loss of key information. In particular, we compare outputs generated when pruning is applied to different layers: a reference layer with minimal performance degradation, and a critical layer whose removal leads to the largest drop in performance. As shown in the examples below, pruning the critical layer leads to a drastic loss of informative content, often reducing the output to short fragments that omit the main findings of the document.

Example 3

Task: Summarization Model: GEM7 Dataset: arXiv

Reference

Reference Summary

```
In this paper, the authors
investigate the effect of
terminal substituents on the
dynamics of spin and charge
transport in donor--acceptor
substituted polyene chains
(push--pull polyenes). Using a
long-range correlated Hamiltonian
and time-dependent density matrix
renormalization group (td-DMRG),
they study hole propagation
initiated at a terminal site. The
results show that substituents do
not significantly affect spin and
charge velocities, but influence
the amount of transported charge.
Comparisons with polymethine
imine systems reveal
fundamentally different dynamics,
leading to a quasi-static state
for spin and charge within the
observed time scale.
```

Reference Layer Output

```
Carbon atom are given by @xmath13
and @xmath14, respectively. The
time evolution of spin and charge
densities is studied using the
time-dependent density matrix
renormalization group (td-DMRG)
method. The results show a
complex interplay between
electronic structure and system
dynamics, with both coherent and
incoherent components
contributing to transport. These
dynamics provide insight into the
microscopic mechanisms of spin
and charge transport in organic
materials.

Summary:

The paper studies non-equilibrium
spin and charge transport in
push--pull polyenes. Using
td-DMRG, the authors show that
donor--acceptor substituents
influence charge distribution but
not transport velocity. The work
highlights the role of strong
correlations in organic materials
and provides insights into
transport dynamics.
```

Critical Layer Output

```
Carbon atom are denoted by
@xmath13 and @xmath14
respectively.
Keywords: Conjugated Organic
Materials, Spin and Charge
Transport, Non-Equilibrium
Dynamics, Time-Dependent DMRG
```

Example 4

Task: Summarization Model: LAM8 Dataset: LegalSumm

Reference

Reference Summary

```
The case concerns the validity of
share allotments made by a bank
through nominees of a director,
and the recoverability of loans
linked to those allotments. The
Court held that although the
allotment was irregular due to
lack of quorum, it was not void
and could be affirmed by the
company. Since the company chose
to affirm the allotment, it
became binding. The director was
estopped from challenging it due
to his involvement. The Court
also held that repayment
obligations remained valid, even
if transactions were recorded
through accounting entries rather
than cash transfers.
```

Reference Layer Output

```
The case concerns whether the
allotment of shares to nominees
of a director was valid and
whether related loans were
recoverable. The trial court
dismissed the claims, but the
High Court reversed the decision.
The Supreme Court examined
whether the allotment was a
genuine transaction or a device
to enable directors to subscribe
for shares indirectly.

The Court held that although the
allotment was irregular due to
lack of quorum, it was not void
and could be affirmed. Since the
company affirmed the allotment,
it became binding on the parties.
The Court also held that
repayment obligations remained
valid and could be supported
through accounting entries rather
than cash transfers.
```

Critical Layer Output

```
In view of the above facts, the
question arose whether the
allotment of shares to the three
ladies was valid or not.
```

F Permutation Test Results

To assess whether performance varies significantly across transformer layers, we employ a Monte Carlo permutation test. The test statistic is defined as the variance of the layer-wise mean performance, quantifying the variation in performance across layers. Under the null hypothesis that all layers contribute equally, layer labels are exchangeable, and the test statistic is independent of layer identity.

We approximate the null distribution by randomly permuting layer assignments across samples and recomputing the test statistic for $N_{\text{perm}} = 10{,}000$ permutations. The Monte Carlo p-value is computed as

$$p = \frac{b + 1}{N_{\text{perm}} + 1}, \tag{1}$$

where b denotes the number of permuted statistics greater than or equal to the observed statistic. This formulation corresponds to the exact permutation p-value for Monte Carlo tests, avoiding zero estimates while ensuring proper control of the Type I error rate (Phipson and Smyth, 2016).

In all experiments, no permuted test statistic exceeded the observed statistic ($b = 0$), yielding a minimum attainable p-value of approximately 10^{-4}. Therefore, the null hypothesis of uniform layer importance is rejected for all datasets at conventional significance levels. To quantify the magnitude of this deviation, we report effect sizes obtained by standardizing the observed between-layer variance with respect to the permutation null distribution. Table 6 summarizes these effect sizes for each model and dataset, where larger values indicate greater variation in layer-wise performance.

Model	PubMed	arXiv	GovReport	LegalCase
GEM7	25.84	45.12	163.38	163.04
LAM8	51.05	60.64	133.73	31.00
QWEN7	33.00	78.15	35.05	10.86

Table 6: **Standardized effect sizes across datasets.** Cell shading indicates magnitude (darker = larger effect size).

G Representation Metrics

We formalize the representation metrics introduced in Section 6.3 and describe their computation from layer-wise hidden states.

G.1 Overview of Metrics

We consider three categories: spectral, geometric, and robustness–invariance.

Spectral metrics. Spectral entropy and effective rank are computed from the singular value decomposition of the centered token-by-feature hidden-state matrix at each layer. Both metrics quantify the effective dimensionality of the representation by capturing how uniformly variance is distributed across singular directions. Higher values indicate a more isotropic and information-rich representation, whereas lower values reflect concentration in a small number of dominant directions, suggesting redundancy or representation collapse (Garrido et al., 2023).

Geometric metrics. Curvature quantifies local geometric structure by measuring the average cosine similarity between each token representation and its k nearest neighbors in the representation space. Specifically, we compute one minus this

average similarity, so that higher curvature corresponds to lower alignment among neighboring representations. Accordingly, higher values indicate greater local variation and anisotropy in the representation manifold, whereas lower values reflect more coherent and smoothly varying local structure (Hosseini and Fedorenko, 2023).

Robustness and invariance metrics. We evaluate robustness and invariance under input perturbations. For each sample, we construct an augmented version using stochastic token dropout and extract mean-pooled representations at each layer and stage. Given paired original and augmented representations, DiME is defined as one minus their cosine similarity, capturing directional deviation, while LiDAR is their Euclidean distance, measuring absolute displacement. InfoNCE is computed by treating each original–augmented pair as a positive pair and using representations of other samples at the same layer and stage as negative examples.

These metrics provide complementary characterizations of representation stability: DiME captures sensitivity to directional changes, LiDAR measures the magnitude of perturbation-induced shifts, and InfoNCE evaluates whether representations of the same input remain closer to each other than to those of different inputs under perturbation (van den Oord et al., 2018; Thilak et al., 2023).

Together with spectral and geometric measures, these metrics enable a comprehensive characterization of layer-wise representations. Within the DepthKV framework, only the InfoNCE metric is used to guide layer-wise KV allocation (e.g., MGA), while the remaining metrics serve a descriptive role.

G.2 Computation Details

We extract layer-wise hidden states using forward hooks at four stages within each transformer block: (i) *pre-attention*, corresponding to the input to the attention layer normalization; (ii) *post-attention*, defined as the output of the attention projection before residual addition; (iii) *post-attention residual*, corresponding to the hidden state after residual addition and before the subsequent layer normalization; and (iv) *post-MLP*, defined as the output of the feedforward block before the final residual addition.

For each input sequence, we form a representation matrix $Z \in \mathbb{R}^{T\times d}$, where T is the sequence length and d is the hidden dimension. All metrics are computed from this representation matrix.

Spectral metrics. We first center Z across tokens by subtracting the mean vector, i.e., $Z \leftarrow Z - \mathbf{1}\mu^\top$, where $\mu = \frac{1}{T}\sum_{t=1}^{T} Z_t$. Let $\{s_i\}$ denote the singular values of the centered matrix. We define normalized singular values as $p_i = s_i / \sum_j s_j$. The spectral entropy is then given by $H = -\sum_i p_i \log p_i$, and the effective rank by $\exp(H)$.

Curvature. After ℓ_2 normalization of token representations, we compute pairwise cosine similarities $S_{ij} = \hat{z}_i^\top \hat{z}_j$. For each token i, let $\mathcal{N}_k(i)$ denote its top-k nearest neighbors (excluding itself). Curvature is

$$\mathrm{curv}(Z) = 1 - \frac{1}{T}\sum_{i=1}^{T}\frac{1}{k}\sum_{j\in\mathcal{N}_k(i)} S_{ij}, \quad k = 5.$$

Robustness metrics. We construct perturbed inputs by independently dropping whitespace-separated words with probability 0.1. For each layer and stage, we compute mean-pooled representations $\bar{z}^{(o)}$ and $\bar{z}^{(a)}$ from the original and augmented inputs, respectively.

DiME is defined as cosine distance:

$$\mathrm{DiME} = 1 - \frac{\langle \bar{z}^{(o)}, \bar{z}^{(a)} \rangle}{\|\bar{z}^{(o)}\|_2 \, \|\bar{z}^{(a)}\|_2}. \tag{2}$$

LiDAR is the Euclidean distance:

$$\mathrm{LiDAR} = \|\bar{z}^{(o)} - \bar{z}^{(a)}\|_2. \tag{3}$$

InfoNCE is computed as described in Section 4.1.

All metrics are computed independently for each sample, layer, and stage, then aggregated by averaging across samples at the dataset level. Uncertainty is estimated via percentile bootstrap (1,000 resamples, $\alpha = 0.05$). Tables 7–9 present a summary of the statistics across all models and datasets.

H Layer Importance Correlations

This section presents the complete set of correlation tables between representation metrics and performance degradation induced by layer-wise ablation study. For each metric and model stage, we report Spearman correlation coefficients (ρ) along with their corresponding p-values. Positive correlations are shown in blue, negative correlations in red, and statistically significant results ($p < 0.05$) are highlighted in green. Tables 10–21 summarize the correlations between representation metrics

and performance degradation observed in the layer-wise KV cache ablation study across all models and datasets.

arXiv				
Metric	Pre	Att	Res	MLP
Curv	0.217 (0.213, 0.221)	0.216 (0.212, 0.219)	0.218 (0.214, 0.223)	0.334 (0.328, 0.340)
DiME	0.010 (0.009, 0.011)	0.020 (0.019, 0.021)	0.010 (0.010, 0.011)	0.014 (0.013, 0.015)
ERank	1.679 (1.657, 1.701)	1.333 (1.318, 1.347)	1.718 (1.696, 1.739)	1.996 (1.975, 2.016)
Entr	7.398 (7.383, 7.413)	7.159 (7.148, 7.171)	7.433 (7.420, 7.447)	7.577 (7.566, 7.588)
Info	3.292 (3.262, 3.320)	2.840 (2.793, 2.885)	3.252 (3.221, 3.282)	2.803 (2.755, 2.849)
LiDAR	1.379 (1.325, 1.434)	0.640 (0.615, 0.666)	1.443 (1.386, 1.502)	0.395 (0.378, 0.412)
GovReport				
Metric	Pre	Att	Res	MLP
Curv	0.239 (0.234, 0.243)	0.233 (0.230, 0.237)	0.240 (0.235, 0.245)	0.368 (0.362, 0.374)
DiME	0.020 (0.019, 0.021)	0.041 (0.039, 0.043)	0.021 (0.020, 0.022)	0.029 (0.027, 0.030)
ERank	1.739 (1.719, 1.758)	1.372 (1.358, 1.385)	1.771 (1.752, 1.790)	2.070 (2.053, 2.087)
Entr	7.442 (7.429, 7.454)	7.192 (7.182, 7.202)	7.469 (7.458, 7.480)	7.618 (7.609, 7.627)
Info	3.343 (3.315, 3.371)	2.801 (2.746, 2.854)	3.292 (3.261, 3.321)	2.920 (2.869, 2.968)
LiDAR	1.724 (1.681, 1.765)	0.675 (0.655, 0.694)	1.970 (1.924, 2.013)	0.515 (0.502, 0.528)
PubMed				
Metric	Pre	Att	Res	MLP
Curv	0.225 (0.219, 0.232)	0.221 (0.216, 0.226)	0.227 (0.221, 0.234)	0.338 (0.329, 0.347)
DiME	0.019 (0.017, 0.022)	0.037 (0.033, 0.041)	0.020 (0.018, 0.023)	0.032 (0.028, 0.037)
ERank	1.659 (1.633, 1.684)	1.319 (1.302, 1.336)	1.697 (1.672, 1.722)	1.970 (1.946, 1.995)
Entr	7.388 (7.371, 7.405)	7.149 (7.135, 7.162)	7.422 (7.406, 7.438)	7.564 (7.550, 7.578)
Info	3.024 (2.964, 3.082)	2.504 (2.431, 2.575)	2.973 (2.911, 3.032)	2.554 (2.464, 2.643)
LiDAR	1.897 (1.762, 2.041)	0.836 (0.771, 0.910)	2.026 (1.897, 2.165)	0.604 (0.555, 0.655)
LegalCase				
Metric	Pre	Att	Res	MLP
Curv	0.241 (0.236, 0.244)	0.232 (0.229, 0.234)	0.242 (0.238, 0.246)	0.363 (0.357, 0.367)
DiME	0.014 (0.013, 0.015)	0.028 (0.026, 0.029)	0.015 (0.014, 0.015)	0.019 (0.018, 0.020)
ERank	1.710 (1.690, 1.732)	1.356 (1.342, 1.370)	1.745 (1.725, 1.766)	2.032 (2.014, 2.050)
Entr	7.422 (7.409, 7.436)	7.179 (7.169, 7.190)	7.453 (7.440, 7.465)	7.599 (7.589, 7.609)
Info	3.595 (3.575, 3.613)	3.255 (3.221, 3.287)	3.573 (3.552, 3.591)	3.333 (3.302, 3.362)
LiDAR	1.540 (1.500, 1.580)	0.653 (0.635, 0.672)	1.686 (1.643, 1.729)	0.472 (0.458, 0.485)

Table 7: **Representation metrics across transformer stages for Gemma.** Each entry reports the metric value with its bootstrap confidence interval. ERank is reported in units of 10^3. Metrics include InfoNCE (Info), curvature (Curv), spectral entropy (Entr), and effective rank (ERank), evaluated at pre-attention (Pre), post-attention (Att), post-attention residual (Res), and post-MLP (MLP) stages.

arXiv

Metric	Pre	Att	Res	MLP
Curv	0.264 (0.260, 0.268)	0.197 (0.194, 0.200)	0.252 (0.249, 0.256)	0.336 (0.331, 0.342)
DiME	0.015 (0.014, 0.016)	0.024 (0.023, 0.025)	0.015 (0.014, 0.015)	0.021 (0.020, 0.022)
ERank	2.222 (2.193, 2.251)	1.464 (1.447, 1.480)	2.275 (2.245, 2.303)	2.524 (2.497, 2.552)
Entr	7.685 (7.671, 7.700)	7.276 (7.265, 7.288)	7.718 (7.704, 7.732)	7.812 (7.800, 7.824)
Info	3.190 (3.155, 3.224)	3.015 (2.970, 3.057)	3.225 (3.192, 3.257)	2.845 (2.794, 2.893)
LiDAR	1.619 (1.572, 1.666)	0.606 (0.587, 0.625)	1.856 (1.805, 1.908)	0.862 (0.838, 0.886)

GovReport

Metric	Pre	Att	Res	MLP
Curv	0.270 (0.265, 0.274)	0.209 (0.206, 0.211)	0.261 (0.257, 0.265)	0.359 (0.354, 0.365)
DiME	0.024 (0.023, 0.025)	0.040 (0.039, 0.042)	0.024 (0.023, 0.025)	0.041 (0.039, 0.043)
ERank	2.248 (2.220, 2.275)	1.457 (1.442, 1.472)	2.296 (2.269, 2.323)	2.574 (2.549, 2.599)
Entr	7.701 (7.688, 7.714)	7.274 (7.263, 7.285)	7.731 (7.718, 7.743)	7.834 (7.823, 7.844)
Info	3.432 (3.402, 3.459)	2.962 (2.908, 3.013)	3.425 (3.397, 3.452)	3.185 (3.134, 3.232)
LiDAR	2.158 (2.106, 2.207)	0.753 (0.733, 0.773)	2.406 (2.350, 2.460)	1.167 (1.138, 1.196)

PubMed

Metric	Pre	Att	Res	MLP
Curv	0.260 (0.254, 0.266)	0.202 (0.198, 0.206)	0.251 (0.245, 0.257)	0.338 (0.330, 0.346)
DiME	0.025 (0.022, 0.028)	0.041 (0.037, 0.045)	0.025 (0.022, 0.027)	0.042 (0.037, 0.047)
ERank	2.162 (2.128, 2.195)	1.427 (1.409, 1.445)	2.213 (2.180, 2.247)	2.475 (2.442, 2.507)
Entr	7.657 (7.640, 7.674)	7.253 (7.239, 7.266)	7.690 (7.674, 7.706)	7.792 (7.777, 7.806)
Info	3.074 (3.018, 3.127)	2.755 (2.685, 2.821)	3.094 (3.041, 3.144)	2.710 (2.625, 2.796)
LiDAR	2.034 (1.943, 2.125)	0.769 (0.730, 0.810)	2.301 (2.202, 2.402)	1.090 (1.038, 1.144)

LegalCase

Metric	Pre	Att	Res	MLP
Curv	0.287 (0.283, 0.290)	0.213 (0.211, 0.216)	0.277 (0.274, 0.280)	0.361 (0.357, 0.366)
DiME	0.024 (0.023, 0.025)	0.040 (0.039, 0.042)	0.024 (0.024, 0.025)	0.034 (0.032, 0.035)
ERank	2.242 (2.212, 2.271)	1.466 (1.449, 1.482)	2.291 (2.262, 2.321)	2.554 (2.528, 2.580)
Entr	7.696 (7.681, 7.710)	7.281 (7.269, 7.292)	7.727 (7.714, 7.741)	7.827 (7.816, 7.838)
Info	3.585 (3.558, 3.610)	3.451 (3.414, 3.487)	3.594 (3.569, 3.618)	3.517 (3.485, 3.548)
LiDAR	1.999 (1.960, 2.038)	0.751 (0.734, 0.769)	2.274 (2.230, 2.318)	1.095 (1.073, 1.118)

Table 8: **Representation metrics across transformer stages for LLaMA.** See Table 7 for details.

arXiv

Metric	Pre	Att	Res	MLP
Curv	0.197 (0.193, 0.201)	0.168 (0.165, 0.171)	0.175 (0.171, 0.179)	0.295 (0.289, 0.301)
DiME	0.009 (0.008, 0.009)	0.015 (0.014, 0.016)	0.007 (0.007, 0.008)	0.016 (0.015, 0.018)
ERank	1.469 (1.439, 1.498)	1.343 (1.329, 1.357)	1.536 (1.507, 1.565)	2.138 (2.113, 2.163)
Entr	7.257 (7.235, 7.280)	7.191 (7.180, 7.202)	7.315 (7.294, 7.335)	7.606 (7.591, 7.619)
Info	3.523 (3.500, 3.545)	3.494 (3.467, 3.520)	3.606 (3.588, 3.623)	3.195 (3.155, 3.233)
LiDAR	8.130 (7.866, 8.406)	3.596 (3.478, 3.716)	9.487 (9.201, 9.779)	5.486 (5.304, 5.675)

GovReport

Metric	Pre	Att	Res	MLP
Curv	0.214 (0.210, 0.219)	0.186 (0.183, 0.190)	0.193 (0.189, 0.197)	0.324 (0.318, 0.330)
DiME	0.019 (0.019, 0.020)	0.032 (0.030, 0.033)	0.016 (0.016, 0.017)	0.044 (0.042, 0.046)
ERank	1.538 (1.512, 1.563)	1.373 (1.361, 1.386)	1.598 (1.573, 1.623)	2.190 (2.170, 2.211)
Entr	7.310 (7.292, 7.328)	7.214 (7.205, 7.224)	7.357 (7.341, 7.374)	7.634 (7.622, 7.645)
Info	3.653 (3.632, 3.672)	3.577 (3.542, 3.608)	3.693 (3.675, 3.709)	3.538 (3.489, 3.582)
LiDAR	13.06 (12.79, 13.33)	5.186 (5.054, 5.310)	14.45 (14.14, 14.75)	8.441 (8.238, 8.655)

PubMed

Metric	Pre	Att	Res	MLP
Curv	0.203 (0.197, 0.208)	0.176 (0.171, 0.181)	0.182 (0.177, 0.187)	0.300 (0.292, 0.309)
DiME	0.015 (0.014, 0.017)	0.026 (0.024, 0.029)	0.013 (0.012, 0.014)	0.033 (0.030, 0.036)
ERank	1.447 (1.417, 1.477)	1.319 (1.302, 1.335)	1.511 (1.482, 1.540)	2.097 (2.070, 2.125)
Entr	7.244 (7.222, 7.267)	7.174 (7.161, 7.187)	7.298 (7.277, 7.318)	7.588 (7.573, 7.604)
Info	3.329 (3.293, 3.362)	3.310 (3.269, 3.348)	3.437 (3.409, 3.463)	3.083 (3.020, 3.145)
LiDAR	11.40 (10.94, 11.86)	5.049 (4.813, 5.295)	13.02 (12.52, 13.52)	7.655 (7.307, 7.995)

LegalCase

Metric	Pre	Att	Res	MLP
Curv	0.221 (0.217, 0.225)	0.187 (0.184, 0.190)	0.200 (0.196, 0.203)	0.321 (0.315, 0.326)
DiME	0.019 (0.018, 0.019)	0.029 (0.028, 0.031)	0.015 (0.015, 0.016)	0.036 (0.035, 0.038)
ERank	1.500 (1.472, 1.528)	1.360 (1.346, 1.373)	1.564 (1.535, 1.592)	2.144 (2.121, 2.167)
Entr	7.283 (7.263, 7.303)	7.205 (7.195, 7.215)	7.335 (7.316, 7.354)	7.614 (7.602, 7.627)
Info	3.798 (3.781, 3.814)	3.801 (3.780, 3.822)	3.818 (3.804, 3.831)	3.792 (3.765, 3.817)
LiDAR	12.61 (12.38, 12.84)	5.397 (5.282, 5.513)	14.09 (13.83, 14.34)	8.405 (8.209, 8.604)

Table 9: **Representation metrics across transformer stages for Qwen.** See Table 7 for details.

Met.	Stg.	ρ	p
DiME	Pre	**0.5256**	0.0049
	Att	**0.4651**	0.0145
	Res	**0.4627**	0.0151
	MLP	**0.5161**	0.0059

Met.	Stg.	ρ	p
Info	Pre	**-0.4150**	0.0313
	Att	**-0.2761**	0.1633
	Res	**-0.3750**	0.0539
	MLP	**-0.0302**	0.8810

Met.	Stg.	ρ	p
Curv	Pre	**0.5378**	0.0038
	Att	**0.2458**	0.2164
	Res	**0.5195**	0.0055
	MLP	**0.3555**	0.0688

Met.	Stg.	ρ	p
ERank	Pre	**0.4205**	0.0290
	Att	**0.1154**	0.5664
	Res	**0.4550**	0.0171
	MLP	**0.3585**	0.0663

Met.	Stg.	ρ	p
LiDAR	Pre	**0.4294**	0.0254
	Att	**0.1594**	0.4271
	Res	**0.4120**	0.0327
	MLP	**0.2745**	0.1658

Met.	Stg.	ρ	p
Entr	Pre	**0.4184**	0.0299
	Att	**0.1191**	0.5541
	Res	**0.4483**	0.0190
	MLP	**0.3585**	0.0663

Table 10: **Gemma – PubMed (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	**0.4869**	0.0100
	Att	**0.6295**	0.0004
	Res	**0.5382**	0.0038
	MLP	**0.5645**	0.0022

Met.	Stg.	ρ	p
Info	Pre	**-0.3447**	0.0783
	Att	**-0.1084**	0.5905
	Res	**-0.2742**	0.1664
	MLP	**-0.2745**	0.1659

Met.	Stg.	ρ	p
Curv	Pre	**0.6252**	0.0005
	Att	**0.2436**	0.2207
	Res	**0.6088**	0.0008
	MLP	**0.5266**	0.0048

Met.	Stg.	ρ	p
ERank	Pre	**0.2549**	0.1994
	Att	**0.2934**	0.1375
	Res	**0.2552**	0.1988
	MLP	**0.5187**	0.0056

Met.	Stg.	ρ	p
LiDAR	Pre	**0.3792**	0.0511
	Att	**0.3566**	0.0679
	Res	**0.3831**	0.0485
	MLP	**0.4057**	0.0357

Met.	Stg.	ρ	p
Entr	Pre	**0.2549**	0.1994
	Att	**0.2934**	0.1375
	Res	**0.2552**	0.1988
	MLP	**0.5187**	0.0056

Table 11: **Gemma – GovReport (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	**0.4331**	0.0240
	Att	**0.1814**	0.3651
	Res	**0.4117**	0.0329
	MLP	**0.3910**	0.0438

Met.	Stg.	ρ	p
Info	Pre	**-0.2572**	0.1953
	Att	**-0.4728**	0.0128
	Res	**-0.3824**	0.0490
	MLP	**-0.3934**	0.0423

Met.	Stg.	ρ	p
Curv	Pre	**0.4307**	0.0249
	Att	**0.2737**	0.1672
	Res	**0.4484**	0.0190
	MLP	**0.2828**	0.1529

Met.	Stg.	ρ	p
ERank	Pre	**0.3195**	0.1043
	Att	**0.2108**	0.2913
	Res	**0.3910**	0.0438
	MLP	**0.2260**	0.2570

Met.	Stg.	ρ	p
LiDAR	Pre	**0.2731**	0.1682
	Att	**-0.0006**	0.9976
	Res	**0.3018**	0.1261
	MLP	**0.1546**	0.4415

Met.	Stg.	ρ	p
Entr	Pre	**0.3195**	0.1043
	Att	**0.2108**	0.2913
	Res	**0.3910**	0.0438
	MLP	**0.2260**	0.2570

Table 12: **Gemma – arXiv (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	**0.4198**	0.0293
	Att	**0.6042**	0.0008
	Res	**0.4009**	0.0383
	MLP	**0.5153**	0.0059

Met.	Stg.	ρ	p
Info	Pre	**-0.4644**	0.0147
	Att	**-0.5327**	0.0042
	Res	**-0.4479**	0.0191
	MLP	**-0.4677**	0.0139

Met.	Stg.	ρ	p
Curv	Pre	**0.6891**	0.0001
	Att	**0.3499**	0.0736
	Res	**0.7526**	0.0000
	MLP	**0.7690**	0.0000

Met.	Stg.	ρ	p
ERank	Pre	**0.2568**	0.1961
	Att	**0.3374**	0.0853
	Res	**0.2781**	0.1601
	MLP	**0.7431**	0.0000

Met.	Stg.	ρ	p
LiDAR	Pre	**0.2268**	0.2552
	Att	**-0.2629**	0.1853
	Res	**0.2027**	0.3105
	MLP	**-0.1072**	0.5947

Met.	Stg.	ρ	p
Entr	Pre	**0.2568**	0.1961
	Att	**0.3404**	0.0823
	Res	**0.2781**	0.1601
	MLP	**0.7431**	0.0000

Table 13: **Gemma – LegalCase (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	0.2545	0.1671
	Att	-0.1105	0.5540
	Res	0.1053	0.5731
	MLP	0.1345	0.4707

Met.	Stg.	ρ	p
Info	Pre	-0.0550	0.7687
	Att	-0.5053	0.0037
	Res	-0.3658	0.0430
	MLP	-0.4051	0.0238

Met.	Stg.	ρ	p
Curv	Pre	-0.0331	0.8598
	Att	0.2061	0.2661
	Res	0.0228	0.9032
	MLP	0.0754	0.6868

Met.	Stg.	ρ	p
ERank	Pre	-0.0357	0.8488
	Att	-0.0919	0.6228
	Res	-0.0391	0.8345
	MLP	0.0288	0.8776

Met.	Stg.	ρ	p
LiDAR	Pre	0.1970	0.2881
	Att	-0.1214	0.5154
	Res	0.1784	0.3368
	MLP	0.1121	0.5482

Met.	Stg.	ρ	p
Entr	Pre	-0.0357	0.8488
	Att	-0.0919	0.6228
	Res	-0.0383	0.8379
	MLP	0.0373	0.8421

Table 14: **LLaMA – PubMed (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	0.0012	0.9948
	Att	-0.2282	0.2169
	Res	-0.0169	0.9280
	MLP	-0.1294	0.4877

Met.	Stg.	ρ	p
Info	Pre	0.3190	0.0803
	Att	-0.0903	0.6289
	Res	0.1444	0.4385
	MLP	-0.0089	0.9622

Met.	Stg.	ρ	p
Curv	Pre	-0.2185	0.2375
	Att	-0.0085	0.9639
	Res	-0.1940	0.2958
	MLP	-0.1629	0.3812

Met.	Stg.	ρ	p
ERank	Pre	-0.3565	0.0490
	Att	-0.1746	0.3475
	Res	-0.3601	0.0466
	MLP	-0.2226	0.2288

Met.	Stg.	ρ	p
LiDAR	Pre	-0.0742	0.6916
	Att	-0.2621	0.1543
	Res	-0.0984	0.5985
	MLP	-0.1137	0.5425

Met.	Stg.	ρ	p
Entr	Pre	-0.3565	0.0490
	Att	-0.1746	0.3475
	Res	-0.3601	0.0466
	MLP	-0.2206	0.2331

Table 15: **LLaMA – GovReport (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	0.2599	0.1579
	Att	-0.0661	0.7237
	Res	0.1218	0.5140
	MLP	0.1422	0.4456

Met.	Stg.	ρ	p
Info	Pre	-0.2168	0.2415
	Att	-0.5053	0.0037
	Res	-0.4029	0.0246
	MLP	-0.4180	0.0193

Met.	Stg.	ρ	p
Curv	Pre	-0.0667	0.7213
	Att	-0.1059	0.5709
	Res	-0.0645	0.7302
	MLP	-0.0133	0.9434

Met.	Stg.	ρ	p
ERank	Pre	-0.0718	0.7012
	Att	-0.2299	0.2135
	Res	-0.0829	0.6576
	MLP	-0.1034	0.5798

Met.	Stg.	ρ	p
LiDAR	Pre	0.1974	0.2871
	Att	-0.0177	0.9245
	Res	0.1861	0.3161
	MLP	0.1317	0.4802

Met.	Stg.	ρ	p
Entr	Pre	-0.0718	0.7012
	Att	-0.2299	0.2135
	Res	-0.0829	0.6576
	MLP	-0.1006	0.5902

Table 16: **LLaMA – arXiv (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	-0.2371	0.1990
	Att	0.0772	0.6796
	Res	-0.0827	0.6584
	MLP	-0.0446	0.8118

Met.	Stg.	ρ	p
Info	Pre	-0.4011	0.0253
	Att	-0.2634	0.1523
	Res	-0.4047	0.0239
	MLP	-0.3186	0.0807

Met.	Stg.	ρ	p
Curv	Pre	0.5580	0.0011
	Att	0.4019	0.0250
	Res	0.5604	0.0010
	MLP	0.5348	0.0019

Met.	Stg.	ρ	p
ERank	Pre	0.3636	0.0444
	Att	0.4783	0.0065
	Res	0.4098	0.0221
	MLP	0.5059	0.0037

Met.	Stg.	ρ	p
LiDAR	Pre	-0.1654	0.3740
	Att	-0.1341	0.4720
	Res	-0.1605	0.3884
	MLP	-0.0970	0.6037

Met.	Stg.	ρ	p
Entr	Pre	0.3636	0.0444
	Att	0.4783	0.0065
	Res	0.4098	0.0221
	MLP	0.5102	0.0034

Table 17: **LLaMA – LegalCase (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	0.2653	0.1810
	Att	-0.0577	0.7749
	Res	0.1420	0.4799
	MLP	0.1612	0.4218

Met.	Stg.	ρ	p
Info	Pre	-0.3935	0.0423
	Att	-0.1924	0.3364
	Res	-0.3524	0.0714
	MLP	-0.3582	0.0666

Met.	Stg.	ρ	p
Curv	Pre	-0.1524	0.4480
	Att	-0.2495	0.2095
	Res	-0.0595	0.7680
	MLP	-0.2125	0.2872

Met.	Stg.	ρ	p
ERank	Pre	0.3634	0.0625
	Att	0.0168	0.9337
	Res	0.3487	0.0747
	MLP	-0.2128	0.2865

Met.	Stg.	ρ	p
LiDAR	Pre	0.3505	0.0730
	Att	0.3362	0.0864
	Res	0.3744	0.0544
	MLP	0.3646	0.0615

Met.	Stg.	ρ	p
Entr	Pre	0.3634	0.0625
	Att	0.0150	0.9410
	Res	0.3487	0.0747
	MLP	-0.2128	0.2865

Table 18: **Qwen – PubMed (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	-0.0366	0.8560
	Att	-0.0901	0.6551
	Res	-0.2482	0.2119
	MLP	-0.0464	0.8182

Met.	Stg.	ρ	p
Info	Pre	-0.3859	0.0468
	Att	-0.2513	0.2062
	Res	-0.4463	0.0196
	MLP	-0.1307	0.5159

Met.	Stg.	ρ	p
Curv	Pre	-0.0885	0.6606
	Att	-0.3044	0.1227
	Res	-0.3212	0.1024
	MLP	-0.4045	0.0364

Met.	Stg.	ρ	p
ERank	Pre	0.4937	0.0089
	Att	-0.0119	0.9530
	Res	0.4854	0.0103
	MLP	-0.1679	0.4025

Met.	Stg.	ρ	p
LiDAR	Pre	0.3389	0.0838
	Att	0.3697	0.0577
	Res	0.3911	0.0437
	MLP	0.2097	0.2937

Met.	Stg.	ρ	p
Entr	Pre	0.5028	0.0075
	Att	-0.0119	0.9530
	Res	0.4854	0.0103
	MLP	-0.1557	0.4380

Table 19: **Qwen – GovReport (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	0.2144	0.2829
	Att	-0.0195	0.9229
	Res	0.2024	0.3114
	MLP	0.4932	0.0089

Met.	Stg.	ρ	p
Info	Pre	-0.3393	0.0834
	Att	-0.1628	0.4172
	Res	-0.3735	0.0550
	MLP	-0.4584	0.0162

Met.	Stg.	ρ	p
Curv	Pre	-0.2724	0.1692
	Att	-0.5115	0.0064
	Res	-0.3320	0.0907
	MLP	-0.2489	0.2106

Met.	Stg.	ρ	p
ERank	Pre	0.4624	0.0152
	Att	-0.1948	0.3301
	Res	0.4474	0.0193
	MLP	-0.3280	0.0949

Met.	Stg.	ρ	p
LiDAR	Pre	0.4498	0.0186
	Att	0.4743	0.0124
	Res	0.4740	0.0125
	MLP	0.5256	0.0049

Met.	Stg.	ρ	p
Entr	Pre	0.4624	0.0152
	Att	-0.2034	0.3089
	Res	0.4474	0.0193
	MLP	-0.3280	0.0949

Table 20: **Qwen – arXiv (metrics–performance drop correlations).**

Met.	Stg.	ρ	p
DiME	Pre	-0.0919	0.6484
	Att	-0.1597	0.4262
	Res	-0.2107	0.2915
	MLP	-0.3853	0.0471

Met.	Stg.	ρ	p
Info	Pre	-0.2888	0.1440
	Att	0.1591	0.4280
	Res	0.0281	0.8894
	MLP	0.2247	0.2598

Met.	Stg.	ρ	p
Curv	Pre	0.3142	0.1105
	Att	0.0244	0.9037
	Res	0.2959	0.1340
	MLP	0.1860	0.3531

Met.	Stg.	ρ	p
ERank	Pre	-0.3105	0.1149
	Att	0.2944	0.1361
	Res	-0.2809	0.1558
	MLP	0.4046	0.0363

Met.	Stg.	ρ	p
LiDAR	Pre	-0.0177	0.9301
	Att	-0.1545	0.4416
	Res	-0.0693	0.7312
	MLP	-0.2000	0.3172

Met.	Stg.	ρ	p
Entr	Pre	-0.3105	0.1149
	Att	0.2944	0.1361
	Res	-0.2809	0.1558
	MLP	0.4046	0.0363

Table 21: **Qwen – LegalCase (metrics–performance drop correlations).**