DISTILLING TO HYBRID ATTENTION MODELS VIA KL-GUIDED LAYER SELECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Distilling pretrained softmax attention Transformers into more efficient hybrid architectures that interleave softmax and linear attention layers is a promising approach for improving the inference efficiency of LLMs without requiring expensive pretraining from scratch. A critical factor in the conversion process is layer selection, i.e., deciding on which layers to convert to linear attention variants. This paper describes a simple and efficient recipe for layer selection that uses layer importance scores derived from a small amount of training on generic text data. Once the layers have been selected we use a recent pipeline for the distillation process itself (RADLADS; Goldstein et al., 2025), which consists of attention weight transfer, hidden state alignment, KL-based distribution matching, followed by a small amount of finetuning. We find that this approach is more effective than existing approaches for layer selection, including heuristics that uniformly interleave linear attentions based on a fixed ratio, as well as more involved approaches that rely on specialized diagnostic datasets.

1 INTRODUCTION

Linear attention (Katharopoulos et al., 2020; Peng et al., 2021; Yang et al., 2023, *i.a.*) and state-space models (Gu et al., 2022; Gu & Dao, 2024; Dao & Gu, 2024, *i.a.*) have gained significant traction recently due to their high inference speed and competitive performance. However, most existing pretrained models are still purely based on softmax attention, and pretraining such linear attention models from scratch is resource-intensive. This has motivated the approaches for *cross-architecture* distillation, a process that converts pretrained Transformer checkpoints into more efficient linear attention counterparts (Kasai et al., 2021; Wang et al., 2024; Bick et al., 2025, *i.a.*).

This distillation process involves two key decisions: (1) the student architecture, and (2) the optimal distillation recipe once the architecture has been selected. For the second question, recent work has shown the effectiveness of a multi-stage pipeline over pure continued finetuning approaches (Bick et al., 2025; Goldstein et al., 2025). This pipeline involves an initial stage of per-layer output alignment with an L_2 loss, followed by a second stage of end-to-end knowledge distillation. What student architecture to distill to, however, remains open. Prior efforts to distill Transformers into purely subquadratic models have often resulted in performance degradation (Zhang et al., 2024a;b; Mercat et al., 2024). More recently, models incorporating a sliding window attention (SWA) mechanism have shown surprisingly strong results across various benchmarks (Lan et al., 2025; Zhang et al., 2025). However, these evaluations have primarily focused on knowledge-intensive common-sense reasoning tasks, where in-context recall plays a lesser role. Our empirical

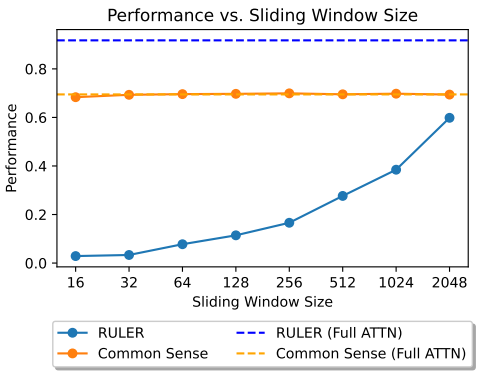


Figure 1: Performance of a sliding-window attention model (distilled from Qwen2.5-3B-Instruct) across different window sizes on RULER and commonsense tasks.

findings show that even a small sliding window of size 16 is sufficient for a distilled SWA model to recover strong performance on such tasks.

In contrast, performance on in-context recall benchmarks like RULER (Hsieh et al., 2024) is highly dependent on the sliding window size (Figure 1). This is perhaps unsurprising, as it reflects the well-documented limitations of fixed-state models in in-context recall (Wen et al., 2025; Arora et al., 2024a;b).

A simple yet effective solution is to incorporate a few global (softmax) attention layers, resulting in a hybrid architecture. This approach has been successfully adopted in recent models pretrained from scratch, such as Jamba (Lenz et al., 2025), MiniMax-01 (MiniMax et al., 2025), Falcon-H1 (Zuo et al., 2025), and Qwen3-Next. These models typically interleave global and linear attention layers at a fixed ratio (e.g., one global layer for every three or seven linear layers) (Wang et al., 2025a). Following this trend, some distillation works have also adopted a fixed interleaving strategy (Wang et al., 2024). However, our preliminary experiments show this uniform approach remains suboptimal for in-context recall, presumably due to the fundamental difference between pretraining and distillation. This observation has been recognized in recent work (Gu et al., 2025; Yang et al., 2025; Hoshino et al., 2025), which also explore various criteria for selectively assigning global attention.

In this work, we adopt a simple global attention selection criterion based on the distillation KL divergence loss: intuitively, the more critical a global attention layer is, the more it reduces the resulting distillation KL loss. Our experiments demonstrate the effectiveness of our selective hybrid distillation, which achieves strong in-context retrieval performance while maintaining efficiency. Our work paves the way for future research on test-time compute scaling for distilled hybrid models (Paliotta et al., 2025; Wang et al., 2025b), where in-context retrieval remains a key bottleneck (Chaudhry et al., 2025).

2 PRELIMINARIES

Notation. Let $\mathbf{X} = [\mathbf{x}_1; \ldots; \mathbf{x}_T] \in \mathbb{R}^{T\times d}$ be a sequence of T token embeddings with model width d. We use L pre-norm Transformer blocks indexed by $\ell \in \{1, \ldots, L\}$, and h attention heads with per-head width d_h so $d = h\, d_h$. A Transformer block then given by

$$\mathbf{U}^{(\ell)} = \mathbf{X}^{(\ell)} + \mathrm{Mix}^{(\ell)}(\mathrm{LN}(\mathbf{X}^{(\ell)})), \qquad \mathbf{X}^{(\ell+1)} = \mathbf{U}^{(\ell)} + \mathrm{FFN}^{(\ell)}(\mathrm{LN}(\mathbf{U}^{(\ell)})).$$

where $\mathrm{Mix}^{\ell}(\cdot)$ is a sequence mixing operation (i.e., softmax or linear attention) for layer ℓ. When not essential, we omit LN and residuals for readability. We write $\mathbf{M}$ for the (additive) attention mask, which encodes causality and any positional encoding (e.g., RoPE/Alibi) as standard.

Softmax attention. For a single head (we suppress head indices) softmax attention proceeds by computing the query, key and value matrices

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V,$$

where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d_h}$ are learnable parameters. The output is given by (with mask $\mathbf{M}$)

$$\mathbf{O} = \mathrm{Softmax}\Big(\tfrac{1}{\sqrt{d_h}}\mathbf{Q}\mathbf{K}^\top + \mathbf{M}\Big)\mathbf{V}, \tag{1}$$

and multi-head concatenates per-head outputs which is transformed by a linear layer $\mathbf{W}_O \in \mathbb{R}^{(hd_h)\times d}$. During autoregressive inference, the same operation admits a recurrent view:

$$\mathbf{o}_t = \sum_{i\le t} \alpha_{t,i}\, \mathbf{v}_i, \qquad \alpha_{t,i} \propto \exp\Big(\tfrac{1}{\sqrt{d_h}}\mathbf{q}_t^\top \mathbf{k}_i\Big), \quad \sum_{i\le t}\alpha_{t,i} = 1. \tag{2}$$

The memory cost of softmax attention grows linearly with respect to sequence length due to the KV cache, which can result in substantial slowdowns as generation length grows due to increasing data movement across the memory hierarchy.

Linear attention. Linear attention layers have been proposed to address the above inefficiencies of softmax attention during decoding. While many variants exist, they generally adopt the following recurrent form:

$$\mathbf{o}_t = \mathbf{q}_t^\top \mathbf{S}_t, \quad \mathbf{S}_t = \mathbf{M}_t\mathbf{S}_{t-1} + \mathbf{k}_t\mathbf{v}_t^\top, \tag{3}$$

where $\mathbf{M}_t$ is a data-dependent and time-varying transition matrix that is a function of $\mathbf{x}_t$. Setting $\mathbf{M}_t = \mathrm{diag}(\boldsymbol{\alpha}_t)$ where $\boldsymbol{\alpha}_t \in \mathbb{R}^d$ is a function of $\mathbf{x}_t$ recovers recent gated linear attention (GLA) variants (Yang et al., 2023; Katsch, 2023; Qin et al., 2024; Peng et al., 2024). Alternatively, using $\mathbf{M}_t = \alpha_t(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^\top)$ recovers the (gated) DeltaNet family of models (Schlag et al., 2021; Yang et al., 2024b;a).[1] The structure of $\mathbf{M}_t$ enables efficient parallel training via a chunking mechanism.

Linear attention compresses the entire history into the hidden state matrix $\mathbf{S}_t$ and thus the memory cost is constant with respect to generation length, leading to much more efficient decoding compared to softmax attention. However, this hidden state bottleneck is a fundamental limitation when it comes to crucial capabilities such as performing associative recall over a given context.

Hybrid attention. A common strategy for maintaining the capabilities of softmax attention while realizing some of the efficiency benefits of linear attention is to use a hybrid model. This approach partitions the set of layer indices into $\mathcal{S}_{\text{softmax}}$ and $\mathcal{S}_{\text{linear}}$ such that $\mathcal{S}_{\text{softmax}} \cup \mathcal{S}_{\text{linear}} = \{1, \ldots, L\}$. Then the sequence-mixing layer is given by

$$\mathrm{Mix}^{(\ell)} = \begin{cases} \mathrm{SoftmaxAttn}^{(\ell)}, & \ell \in \mathcal{S}_{\text{softmax}}, \\ \mathrm{LinearAttn}^{(\ell)}, & \ell \in \mathcal{S}_{\text{linear}}. \end{cases}$$

Recent works have shown that architectures that use a fixed ratio of linear to softmax attention layers performs well when pretrained from scratch (Lenz et al., 2025; MiniMax et al., 2025). However, such a uniform strategy may be suboptimal for distilling hybrid attention models from pretrained softmax attention models, motivating our present work on layer selection for distillation.

3 Layer Selection for Distilling Hybrid Attention

For distilling a pretrained softmax attention LLM into a hybrid attention model, we seek to find a set $\mathcal{L}_{\text{soft}}$ for a given budget $|\mathcal{L}_{\text{soft}}| = K$ such that converting all the other layers into linear attention has minimal performance degradation. Solving this exactly would require a combinatorial search over all possible K-sized subsets of $[L]$, which would be intractable. Our key idea is to measure a layer's *marginal utility* by restoring exactly that layer (and only that layer) to softmax in an otherwise all-linear student, then distilling briefly and scoring how much the teacher–student KL improves.

3.1 Initial distillation to an all-linear student

We first distill to an all-linear student model, adopting the first two stages of the distillation pipeline from RADLADS (Goldstein et al., 2025). Let $\mathcal{M}_{\text{teacher}}$ be the original teacher model and $\mathcal{M}_{\text{all-linear}}$ be an all-linear student model, where the linear attention parameters are initialized from the teacher's parameters, i.e., $(\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V, \mathbf{W}_O)$. The other parameters of the linear attention layer (in particular the parameters of a linear layer for the data-dependent gating term α_t) are initialized randomly. Then distillation proceeds as follows:

Stage 1: Hidden-state alignment. For a given token sequence $\boldsymbol{x} = x_1 \ldots x_T$, the attention hidden states from the all-linear student model $\{\mathbf{U}^{(\ell)}_{\text{all-linear}}\}_{\ell \in [l]}$ are trained to match the teacher's hidden states $\{\mathbf{U}^{(\ell)}_{\text{teacher}}\}_{\ell \in [l]}$,

$$\mathcal{L}_{\text{hidden}}(\mathcal{M}_{\text{all-linear}}, \boldsymbol{x}) = \sum_{\ell \in [L]} \frac{1}{T} \left\| \mathbf{U}^{(\ell)}_{\text{teacher}} - \mathbf{U}^{(\ell)}_{\text{all-linear}} \right\|_2^2. \tag{4}$$

Here, we only train the parameters of the student's linear attention layer while freezing FFN's parameters. The targets are produced by the teacher model and remain fixed.

Stage 2: Distribution matching. In stage 2 we minimize a temperature-scaled KL between teacher logits $\boldsymbol{\ell}_{\text{teacher},t} \in \mathbb{R}^V$ and student logits $\boldsymbol{\ell}_{\text{all-linear},t} \in \mathbb{R}^V$ with respect to all student parameters (i.e., including the student's FFN layers)

$$\mathcal{L}_{\text{KL}}(\mathcal{M}_{\text{all-linear}}, \boldsymbol{x}) = \frac{\tau^2}{T} \sum_{t=1}^{T} \mathrm{KL}\Big(\mathrm{Softmax}\Big(\tfrac{\boldsymbol{\ell}_{\text{teacher,t}}}{\tau}\Big) \,\Big\|\, \mathrm{Softmax}\Big(\tfrac{\boldsymbol{\ell}_{\text{all-linear,t}}}{\tau}\Big) \Big), \tag{5}$$

[1] DeltaNet also multiplies the additive term $\mathbf{k}_t \mathbf{v}_t^\top$ with β_t, which we omit for simplicity.

where τ smoothing term that provides stronger gradient signal on non-argmax tokens. (The functions $\mathcal{L}_{\text{hidden}}$ and $\mathcal{L}_{\text{KL}}$ are obviously functions of $\mathcal{M}_{\text{teacher}}$ but we omit it from the argument for readability.)

Stage 1 uses 100M tokens while stage 2 uses 600M tokens. All subsequent applications of the stagewise pipeline (i.e., in §3.2 and §3.3) use the same number of tokens.[2]

3.2 Deriving Layerwise Importance Scores

With the all-linear model $\mathcal{M}_{\text{all-linear}}$ derived from the above process in hand, we now describe our layer selection strategy. Let $\mathcal{M}^{(-\ell)}_{\text{all-linear}}$ be a model derived from $\mathcal{M}_{\text{all-linear}}$ where the ℓ-th block has been restored back into the ℓ-th layer of $\mathcal{M}_{\text{teacher}}$. We run stage 1 and stage 2 of the above process again to finetune the student $\mathcal{M}^{(-\ell)}_{\text{all-linear}}$, which now has one softmax attention layer. We define $\mathcal{I}(\ell)$, the layer importance for layer ℓ, as the KL divergence between and the teacher model, i.e.,

$$\mathcal{I}(\ell) = -\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}\big[\mathcal{L}_{\text{KD}}(\mathcal{M}^{(-\ell)}_{\text{all-linear}}, \boldsymbol{x})\big]. \tag{6}$$

Higher $\mathcal{I}(\ell)$ means larger KL reduction (i.e., greater marginal utility under our objective). Because the baseline student and neighbors are fixed, $\mathcal{I}(\ell)$ is hybrid-aware and variant-aware.

3.3 Layer Selection and Final Distillation

Algorithm 1 KL-guided Layer Selection for Hybrid Attention Distillation

Require: Teacher $\mathcal{M}_{\text{teacher}}$; dataset $\mathcal{D}$ (DCLM); temperature τ; target budget K
1: Distill into pure linear attention model $\mathcal{M}_{\text{all-linear}}$ (§3.1)
2: **for** $\ell = 1$ to L **in parallel do** (§3.2)
3: Obtain $\mathcal{M}^{(-\ell)}_{\text{all-linear}}$ by changing ℓ-th layer of $\mathcal{M}_{\text{all-linear}}$ to ℓ-th layer of $\mathcal{M}_{\text{teacher}}$
4: **Stage 1:** align all linear blocks by $\mathcal{L}_{\text{hid}}$ on $\mathcal{D}$.
5: **Stage 2:** distill by $\mathcal{L}_{\text{KL}}$ on $\mathcal{D}$.
6: Compute $\mathcal{I}(\ell) = -\mathbb{E}[\mathcal{L}_{\text{KL}}]$ on a held-out slice of $\mathcal{D}$.
7: **end for**
8: **Select:** $\mathcal{S}_{\text{softmax}} \leftarrow$ top-K layers by $\mathcal{I}(\ell)$ (§3.3)
9: **Final hybrid:** instantiate hybrid based on $\mathcal{S}_{\text{softmax}}$ and linear on layers $[L] \setminus \mathcal{S}_{\text{softmax}}$; train with the two-stage distillation pipeline.

Given a budget of K softmax attention layers that we can keep, we now take the top-K most important layers and convert the result into linear attention i.e.,

$$\mathcal{S}_{\text{softmax}} = \text{top-K}(\mathcal{I}(\ell)), \quad \mathcal{S}_{\text{linear}} = \{1, \ldots, L\} \setminus \mathcal{S}_{\text{softmax}}.$$

Denoting the above hybrid model with K softmax attention layers as $\mathcal{M}_{\text{hybrid-}K}$ we run a final distillation pipeline by rerunning stages 1 and 2 with this hybrid model. Our full algorithm is given in Algorithm 1.

4 Experiments

Having introduced our method, we now present a series of experiments designed to build a comprehensive case for its effectiveness. We begin by establishing why hybrid models are essential for maintaining long-context capabilities (§4.1). We then demonstrate that our KL-guided approach outperforms a wide range of baselines (§4.3).

[2]For our main GA-S2 selector, the final hybrid model reuses the Stage 1-aligned linear attention layers from $\mathcal{M}_{\text{all-linear}}$ and therefore only runs Stage 2 in the last distillation step. For heuristic baselines that are not initialized from $\mathcal{M}_{\text{all-linear}}$, we run both Stage 1 and Stage 2 in the final distillation for fairness.

4.1 The Case for Hybrid Models

There has been a flurry of recent work on distilling to pure linear attention models (Chen et al., 2024; Mercat et al., 2024; Zhang et al., 2025; Goldstein et al., 2025; Wang et al., 2024; Yueyu et al., 2025; Lan et al., 2025; Bick et al., 2025). These works generally report that pure linear attention can maintain the performance of pretrained softmax attention baselines with the right distillation process. However, this conclusion is often based on comparing performance on tasks such as MMLU and Commonsense Reasoning, whose context lengths are short; it is unclear the extent to which such pure linear attention models can maintain performance on benchmarks which require understanding and performing recall over longer contexts. To analyze this, we construct a series of hybrid models based on our approach where the number of softmax layers ranges from 1 to $L-1$. We then evaluate these models on RULER (Hsieh et al., 2024), a diagnostic benchmark designed to probe the long-context capabilities of LLMs. We also evaluate these models on short-context commonsense reasoning benchmarks evaluated by previous methods, including PIQA, ARC-Easy, ARC-Challenge, HellaSwag and WinoGrande (we report the average).

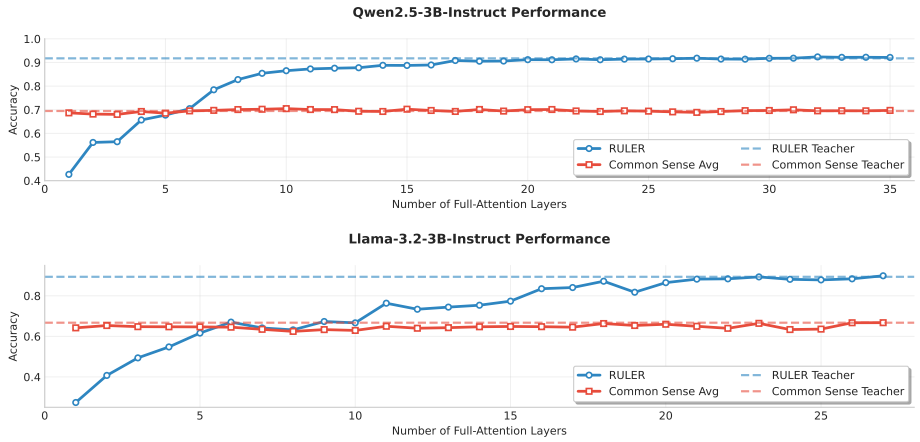


Figure 2: Performance on recall-intensive vs. commonsense tasks as the number of full-attention layers is varied for Qwen2.5-3B-Instruct (top) and Llama-3.2-3B-Instruct (bottom). Recall ability is highly sensitive to the softmax budget, while commonsense reasoning is not.

The results in Figure 2 reveal a stark dichotomy. Performance on the long-context RULER benchmark is highly sensitive to the number of softmax layers (K), growing monotonically and confirming that global context aggregation is critical for in-context retrieval. In contrast, commonsense reasoning performance is almost entirely insensitive to K; models with even a single softmax layer achieve near-teacher-level performance, suggesting these local tasks are well-handled by linear attention. Ironically, the efficiency benefits of linear attention are minimal on precisely these short-context tasks. This dichotomy motivates our work: the central challenge in distilling hybrid models is to preserve long-context recall. This requires a method that can judiciously allocate a limited budget of expensive softmax layers to the positions where they are most impactful.

4.2 Experimental Setup

Having established the importance of selection, we now evaluate our KL-guided method against the a suite of baselines.

Model and data. We evaluate two 3B-class decoder-only teachers: **Qwen2.5-3B-Instruct** and **Llama-3.2-3B-Instruct**. For each architecture we take the checkpoint's native depth L and report K to match the target softmax:linear ratio. We target four ratios 1:8, 1:3, 1:2, 1:1 (thus $K \in \{4, 9, 12, 18\}$ when L=36; if L differs, we use the nearest integer K). All selection and distillation runs use the **DCLM** (Li et al., 2025) generic-text mixture. As noted in § 3.1, each instance of stage 1 uses 100M tokens while stage 2 uses 600M tokens.

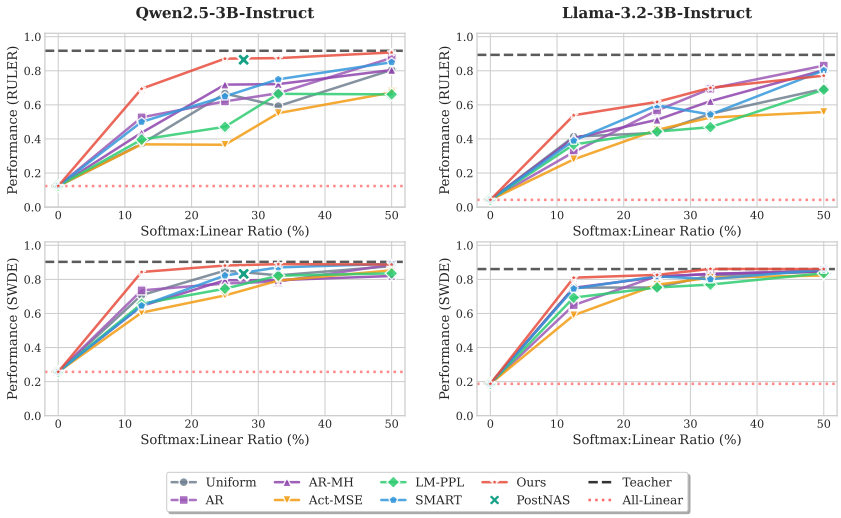


Figure 3: Performance comparison of various layer selection methods on RULER (top) and SWDE (bottom) for distilling Qwen2.5-3B (left) and Llama-3.2-3B (right) into hybrid GDN-based models. Performance is plotted against the percentage of softmax layers retained. The dashed line indicates the performance of the all-softmax teacher model.

Baselines. We compare our one-swap selector to the baselines below. Each returns a set of K softmax layers and is trained with the same two-stage distillation and token budget as ours (§3.1): (1) **Uniform interleave (UNIFORM).** Pick K layers by evenly spacing them across depth (one roughly every $\lfloor L/K \rfloor$ blocks), as adopted by Wang et al. (2024). (2) **Task-guided selectors.** AR (Associative Recall): bypass each layer and measure the drop on a synthetic key–value recall task and then rank layer importance by drop in performance (Chaudhry et al., 2025). AR-MH (Associative Recall - Multihop): same as AR but with multi-hop alias chains, which makes the task more difficult. (3) **Model-signal selectors.** ACT-MSE: layer importance is derived from zero-ing out a layer and measuring increase in activation MSE vs. the baseline. LM-PPL: same as Act-MSE, but derived from measuring an increase in LM perplexity on held-out data. (4) **SMART** (Yang et al., 2025). A sensitivity-aware strategy: (i) score each layer by the reduction in teacher–student KL when swapping an global layer into an otherwise linear baseline; (ii) preserve high-score layers near input/output (so-called "terminal preservation"); (iii) choose the rest from near-uniform candidates to maximize total sensitivity. We also compare against **PostNAS** (Gu et al., 2025), a contemporaneous work that uses a more complex search procedure. Their method involves training a once-for-all SuperNet and then using beam search to find the optimal K softmax layers for a specific downstream task. This process is computationally intensive, requiring 50B training tokens, whereas our selection pipeline uses only 5-6B tokens. Fortunately, PostNAS released their selected layers for the Qwen2.5 model. To ensure a fair comparison, we take their publicly released layer set and distill it using our own pipeline and token budget. More baselines descriptions are included in Table 3 in the Appendix A.

4.3 MAIN RESULTS

We use gated DeltaNet (GDN) for our linear attention layer and evaluate our proposed layer selection method against the baselines for Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct teachers. The results on two long-context, recall-intensive benchmarks, RULER and SWDE, are presented in Figure 3. Our central finding is that our selection method consistently and substantially outperforms all other baselines across both models and tasks. This demonstrates the effectiveness of using a brief, KL-divergence-guided distillation to derive model-intrinsic layer importance scores for creating hybrid architectures.

A key advantage of our approach is particularly evident in the low-budget regime, where only a small fraction of layers are kept as full softmax attention. For instance, on the RULER benchmark with the Qwen2.5 model at a 12.5% ratio (corresponding to 5 attention layers), our method achieves a score of nearly 0.70, whereas the next best baseline, AR, scores around 0.53, and the common UNIFORM

interleaving strategy scores below 0.40. This pronounced gap at low softmax ratios highlights our method's efficiency in identifying the most critical layers for preserving long-context recall, enabling significant performance gains with minimal computational overhead from expensive attention layers.

As the budget for softmax layers increases, our method continues to maintain a performance advantage, approaching the teacher model's performance more rapidly than competing approaches. For both models, a hybrid with 50% of its layers selected by our method recovers a vast majority of the teacher's performance on these challenging recall tasks. Similar performance trends were observed on other benchmarks, including FDA and SQuADv2; these results are detailed in the Appendix A.

5 Analysis

In this section, we conduct a series of ablation studies to deconstruct our method (§5.1), understand its architectural sensitivities (§5.2), and validate its practical efficiency (§5.3).

5.1 The Importance of KL and Greedy Addition Strategy

Our proposed layer selection method involves two key design choices: (1) we use the stage-2 (S2) knowledge distillation (KL-based) loss as the importance metric for each layer in the one-swap setting of §3.2, and (2) given these layerwise scores, we select the top-K softmax layers in a greedy addition fashion (GA), i.e., we keep the K layers that yield the largest marginal KL reduction relative to the all-linear baseline. There are natural alternatives: we could use the stage-1 (S1) hidden-state alignment (MSE-based) metric as our layer importance; we could also use a greedy *removal* (GR) search strategy, which starts from an all-softmax model and greedily converts the least important layer to a linear attention layer. It is also possible to average the layer importance rankins from both GA and GR (**AVG**). Note that our main proposed method corresponds to GA-S2.

The ablation results, presented in Table 1, show that the Stage-2 (KL-based) methods consistently and dramatically outperform their Stage-1 (MSE-based) counterparts, and our greedy addition strategy (GA-S2) is more effective than greedy removal (GR-S2). This suggests that identifying the single most impactful layer to add from an all-linear base is a more robust signal than identifying the least harmful layer to remove.

	Stage 1 (MSE-based)			**Stage 2 (KL-based)**		
Model	GR-S1	GA-S1	AVG-S1	GR-S2	**GA-S2 (Ours)**	AVG-S2
Llama-3.2-3B-Instruct	0.4508	0.4193	0.4233	0.4950	**0.6174**	0.5580
Qwen2.5-3B-Instruct	0.4827	0.5408	0.4933	0.8259	**0.8713**	0.8205

Table 1: Ablation on layer selection strategies for a fixed 25% softmax ratio. We compare Greedy Addition (GA), Greedy Removal (GR), and Averaged (AVG) search using either a Stage-1 (MSE) or Stage-2 (KL) importance metric.

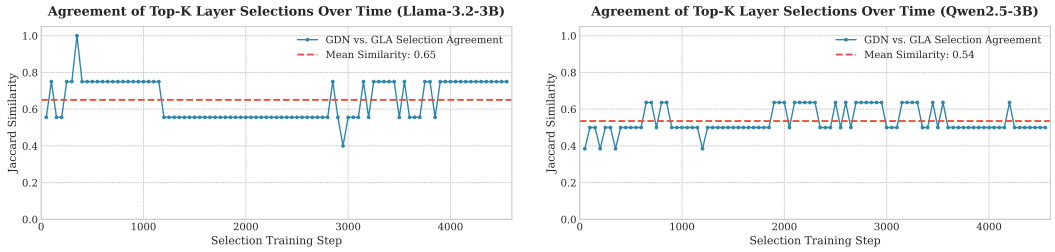
5.2 The Importance of Architecture Consistency

Our layer selection approach is sensitive to the type of linear attention layer employed. To what extent is this selection approach architecture-agnostic—i.e., is our method simply finding a fixed set of "important layers" in the teacher, or is it adapting its selection to the specific architecture of the student's linear layers? To test this, we run the selection process independently for both GDN and GLA students and analyze the results.

	Llama-3.2-3B		**Qwen2.5-3B**	
Ratio	GDN	GLA	GDN	GLA
12.5%	0.5389	0.4918	0.6946	0.5903
25%	0.6174	0.6379	0.8713	0.6921
33%	0.7003	0.7108	0.8743	0.8811
50%	0.7712	0.7644	0.9074	0.8950

Figure 4: Final RULER performance using architecture-specific selections.

The results in Figure 5 and Table 4 reveal an interesting architectural dependence. For **Llama-3.2-3B-Instruct**, the layer selections for GDN



(a) Llama-3.2-3B (Mean Similarity: 0.65) (b) Qwen2.5-3B (Mean Similarity: 0.54)

Figure 5: Jaccard similarity of top-K layer selections between GDN and GLA variants over the selection pass. Llama shows higher agreement, suggesting its layer importance is less student-dependent.

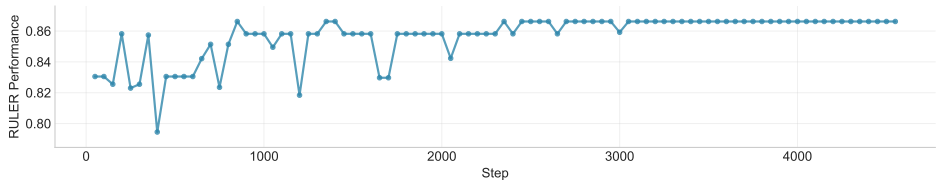


Figure 6: The evolution of RULER performance during the Stage-2 selection process for Qwen2.5-3B-Instruct.

and GLA show high agreement (mean Jaccard similarity of 0.65), and the final models perform almost identically. This suggests that for the Llama architecture, our method identifies a robust, largely student-agnostic set of important layers. For **Qwen2.5-3B-Instruct**, however, the story is more nuanced. The agreement between selections is much lower (mean similarity of 0.54), indicating that the two student variants favor different layers. This divergence has a dramatic impact on performance: the specialized GDN-GDN model (0.8713 on RULER) is vastly superior to the specialized GLA-GLA model (0.6921).

Model	**Student**	UNIFORM	AR	AR-MH	MSE	PPL	SMART	**Ours**
Llama	GDN	0.4359	0.5671	0.5123	0.4534	0.4432	0.5974	**0.6174**
	GLA	0.4050	0.5671	0.5115	0.3983	0.3866	0.5767	**0.6014**
Qwen	GDN	0.6663	0.6203	0.7187	0.3658	0.4712	0.6103	**0.8713**
	GLA	0.6334	0.5689	0.6628	0.3435	0.4296	0.5771	**0.8613**

Table 2: Performance on RULER for GDN- and GLA-based hybrid students at a fixed 25% softmax ratio. For both student variants, the layer set for our method (**Ours**) was selected using a GDN-based process to test for transferability. Note that Llama refers to Llama-3.2-3B-Instruct and Qwen refers to Qwen2.5-3B-Instruct.

Most surprisingly, when we test the transferability by using the GDN-selected layers to distill a **GLA student**, we achieve a RULER score of 0.8613 (Table 2). This result is not only far better than all baselines, but is also significantly better than the score from the specialized GLA-GLA process (0.6921). This reveals a key finding: the choice of linear attention variant used during the selection pass acts as a "probe", and some probes are better than others at identifying a truly robust set of important layers for a given teacher architecture. For the Qwen model, using **GDN as the probe** in our selection algorithm yields a universally superior set of layers that benefits both GDN and GLA students. For the Llama model, both probes are equally effective. This demonstrates that our method's strength is not just in specialization, but in its ability to leverage different student architectures to find the most fundamentally important layers in the teacher.

5.3 HOW MANY TOKENS ARE REALLY NECESSARY FOR LAYER SELECTION?

We used 100M tokens for stage 1 and 600M tokens for stage 2 following the recipe recommended in Goldstein et al. (2025). However, it is possible that the layer selection process could be even more

token-efficient. To investigate this, we tracked the top-K layer set chosen by our selector throughout the Stage-2 training process (at a 1:3 softmax ratio for both models). We measured stability over time using rolling-window Jaccard similarity and the size of the intersection between consecutive sets (the "backbone"). For both teacher models, we find that the set of selected layers stabilizes long before the full training budget is consumed. A nearly complete "backbone" of $K - 1$ layers is typically identified within the first 25-40% of training. Continuing training beyond this point only refines the choice for the final one or two slots, with a negligible impact on the final model's RULER performance (a difference of less than 0.01 absolute points). This observation suggests that a simple stability-based rule can dramatically improve efficiency. For instance, a conservative early stopping point for our runs would have reduced the token budget for the selection pass by 58–74%. The effectiveness of this early stopping rule is backed by our empirical observation: for Qwen, the RULER performance during Stage-2 stabilizes around step 1500, as shown in Figure 6. For more details, please refer to Appendix B.

6 Related Work

In-context recall presents a significant challenge for subquadratic models, a difficulty often attributed to the perplexity gap between them and standard transformers (Arora et al., 2024a). One promising approach to address this is the development of linear attention variants with superior recall capabilities. The seminal work on DeltaNet (Schlag et al., 2021; Yang et al., 2024b) and its successors (Yang et al., 2024a; Siems et al., 2025; Grazzi et al., 2025) has demonstrated great success in this area. Nevertheless, these recurrent approaches are fundamentally limited in associative recall by their fixed-size state (Wen et al., 2025; Arora et al., 2024a). Highlighting the importance of this problem, recent work reveals a connection between in-context recall and test-time scaling performance, arguably making it one of the most critical research directions in efficient sequence model design (Chaudhry et al., 2025). Other notable efforts to improve recall include reading inputs twice (Arora et al., 2024c), dynamic state allocation (Ben-Kish et al., 2025), and dynamic caching for hard-to-memorize items (Nguyen et al., 2025).

Hybrid attention architectures, which combine the complementary strengths of global attention (for accurate retrieval) and linear attention (for fast local processing), can theoretically overcome these state-size limitations (Wen et al., 2025; Arora et al., 2024b). While most hybrid models adopt an inter-layer strategy, interleaving global and linear attention layers (Ren et al., 2025; MiniMax et al., 2025; Lenz et al., 2025), we also note the potential of intra-layer hybridization schemes for efficient time mixing (Irie et al., 2025; Dong et al., 2024; Zuo et al., 2025; Zancato et al., 2024). However, pretraining these linear and hybrid models from scratch is computationally expensive. An effective alternative is to distill a pretrained softmax attention model into a linear attention-based one. This concept was first proposed by Kasai et al. (2021). Subsequent work has emphasized preserving or mimicking the softmax operator during distillation to maintain performance while achieving linear complexity Peng et al. (2022); Zhang et al. (2024b;a). Research work shows that sliding window attention with window size 64 works well in many benchmarks Lan et al. (2025); Zhang et al. (2025), though we show in this work that such strategies still perform poorly on in-context recall.

In the context of distilling into a hybrid of global and linear attention, a key question has emerged: how to select which global attention patterns to preserve. Some methods rely on downstream benchmark performance to determine importance Gu et al. (2025), while others use speculative decoding as a diagnostic tool to identify redundant attention layers Hoshino et al. (2025). In contrast, our work focuses on a simple strategy using an unsupervised learning loss and provides extensive analysis that goes beyond prior research (Yang et al., 2025).

7 Conclusion

In this work, we introduced a simple and effective method for selecting which softmax attention layers to retain when distilling a pretrained Transformer into a more efficient hybrid architecture. While our selection process is more efficient than complex search-based alternatives, future work could explore even cheaper proxies for layer importance, potentially derived directly from the teacher model's activations or gradients. Other promising directions include extending this selection framework from the layer level to a more fine-grained, head-level hybridization.

STATEMENT ON LLM USAGE

We acknowledge the use of Large Language Models (LLMs) to assist in the preparation of this manuscript. Specifically, LLMs were utilized to improve grammar and clarity, aid in literature discovery, and generate boilerplate code snippets for our experiments and testing scripts. The authors have carefully reviewed and edited all LLM-generated outputs and take full responsibility for the final content and scientific integrity of this work.

REFERENCES

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Re. Zoology: Measuring and improving recall in efficient language models. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=LY3ukUANko.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, James Zou, Atri Rudra, and Christopher Re. Simple linear attention language models balance the recall-throughput tradeoff. In *Forty-first International Conference on Machine Learning*, 2024b. URL https://openreview.net/forum?id=e93ffDcpH3.

Simran Arora, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: closing the recall gap for recurrent language models, 2024c. URL https://arxiv.org/abs/2407.05483.

Assaf Ben-Kish, Itamar Zimerman, Muhammad Jehanzeb Mirza, Lior Wolf, James R. Glass, Leonid Karlinsky, and Raja Giryes. Overflow prevention enhances long-context recurrent LLMs. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=h99hJlU99U.

Aviv Bick, Tobias Katsch, Nimit Sohoni, Arjun Desai, and Albert Gu. Llamba: Scaling distilled recurrent models for efficient language processing, 2025. URL https://arxiv.org/abs/2502.14458.

Hamza Tahir Chaudhry, Mohit Kulkarni, and Cengiz Pehlevan. Test-time scaling meets associative memory: Challenges in subquadratic models. In *New Frontiers in Associative Memories*, 2025. URL https://openreview.net/forum?id=QjRZNhfOVL.

Hanting Chen, Zhicheng Liu, Xutao Wang, Yuchuan Tian, and Yunhe Wang. Dijiang: Efficient large language models through compact kernelization. *arXiv preprint arXiv:2403.19928*, 2024.

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

Xin Dong, Yonggan Fu, Shizhe Diao, Wonmin Byeon, Zijia Chen, Ameya Sunil Mahabaleshwarkar, Shih-Yang Liu, Matthijs Van Keirsbilck, Min-Hung Chen, Yoshi Suhara, Yingyan Lin, Jan Kautz, and Pavlo Molchanov. Hymba: A hybrid-head architecture for small language models, 2024. URL https://arxiv.org/abs/2411.13676.

Daniel Goldstein, Eric Alcaide, Janna Lu, and Eugene Cheah. Radlads: Rapid attention distillation to linear attention decoders at scale. *arXiv preprint arXiv:2505.03005*, 2025.

Riccardo Grazzi, Julien Siems, Jörg K.H. Franke, Arber Zela, Frank Hutter, and Massimiliano Pontil. Unlocking state-tracking in linear RNNs through negative eigenvalues. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=UvTo3tVBk2.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Proceedings of CoLM*, 2024.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *Proceedings of ICLR*, 2022.

Yuxian Gu, Qinghao Hu, Shang Yang, Haocheng Xi, Junyu Chen, Song Han, and Han Cai. Jet-nemotron: Efficient language model with post neural architecture search, 2025. URL https://arxiv.org/abs/2508.15884.

Yuichiro Hoshino, Hideyuki Tachibana, Muneyoshi Inahara, and Hiroto Takegawa. Rad: Redundancy-aware distillation for hybrid models via self-speculative decoding, 2025. URL https://arxiv.org/abs/2505.22135.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Kazuki Irie, Morris Yau, and Samuel J. Gershman. Blending complementary memory systems in hybrid quadratic-linear transformers, 2025. URL https://arxiv.org/abs/2506.00744.

Jungo Kasai, Hao Peng, Yizhe Zhang, Dani Yogatama, Gabriel Ilharco, Nikolaos Pappas, Yi Mao, Weizhu Chen, and Noah A. Smith. Finetuning pretrained transformers into RNNs. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10630–10643, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.830. URL https://aclanthology.org/2021.emnlp-main.830/.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of ICML*, 2020.

Tobias Katsch. Gateloop: Fully data-controlled linear recurrence for sequence modeling. *arXiv preprint arXiv:2311.01927*, 2023.

Disen Lan, Weigao Sun, Jiaxi Hu, Jusen Du, and Yu Cheng. Liger: Linearizing large language models to gated recurrent structures. *arXiv preprint arXiv:2503.01496*, 2025.

Barak Lenz, Opher Lieber, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, Daniel Gissin, Daniel Jannai, Dor Muhlgay, Dor Zimberg, Edden M. Gerber, Elad Dolev, Eran Krakovsky, Erez Safahi, Erez Schwartz, Gal Cohen, Gal Shachaf, Haim Rozenblum, Hofit Bata, Ido Blass, Inbal Magar, Itay Dalmedigos, Jhonathan Osin, Julie Fadlon, Maria Rozman, Matan Danos, Michael Gokhman, Mor Zusman, Naama Gidron, Nir Ratner, Noam Gat, Noam Rozen, Oded Fried, Ohad Leshno, Omer Antverg, Omri Abend, Or Dagan, Orit Cohavi, Raz Alon, Ro'i Belson, Roi Cohen, Rom Gilad, Roman Glozman, Shahar Lev, Shai Shalev-Shwartz, Shaked Haim Meirom, Tal Delbari, Tal Ness, Tomer Asida, Tom Ben Gal, Tom Braude, Uriya Pumerantz, Josh Cohen, Yonatan Belinkov, Yuval Globerson, Yuval Peleg Levy, and Yoav Shoham. Jamba: Hybrid transformer-mamba language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=JFPaD7lpBD.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. Datacomp-lm: In search of the next generation of training sets for language models, 2025. URL https://arxiv.org/abs/2406.11794.

Jean Mercat, Igor Vasiljevic, Sedrick Scott Keh, Kushal Arora, Achal Dave, Adrien Gaidon, and Thomas Kollar. Linearizing large language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=soGxskHGox.

MiniMax, Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, Enwei Jiao, Gengxin Li, Guojun Zhang, Haohai Sun, Houze Dong, Jiadai Zhu, Jiaqi Zhuang, Jiayuan Song, Jin Zhu, Jingtao Han, Jingyang Li, Junbin Xie, Junhao Xu, Junjie Yan, Kaishun Zhang, Kecheng Xiao, Kexi Kang, Le Han, Leyang Wang, Lianfei Yu, Liheng Feng, Lin Zheng, Linbo Chai, Long Xing, Meizhi Ju, Mingyuan Chi, Mozhi Zhang, Peikai Huang, Pengcheng Niu, Pengfei Li, Pengyu Zhao, Qi Yang, Qidi Xu, Qiexiang Wang, Qin Wang, Qiuhui Li, Ruitao Leng, Shengmin Shi, Shuqi Yu, Sichen Li, Songquan Zhu, Tao Huang, Tianrun Liang, Weigao Sun, Weixuan Sun, Weiyu Cheng, Wenkai Li, Xiangjun Song, Xiao Su, Xiaodong Han, Xinjie Zhang, Xinzhu Hou, Xu Min, Xun Zou, Xuyang Shen, Yan Gong, Yingjie Zhu, Yipeng Zhou, Yiran Zhong, Yongyi Hu, Yuanxiang Fan, Yue Yu, Yufeng Yang, Yuhao Li, Yunan Huang, Yunji Li, Yunpeng Huang, Yunzhi Xu, Yuxin Mao, Zehan Li, Zekang Li, Zewei Tao, Zewen Ying, Zhaoyang Cong, Zhen Qin, Zhenhua Fan, Zhihang Yu, Zhuo Jiang, and Zijia Wu. Minimax-01: Scaling foundation models with lightning attention, 2025. URL https://arxiv.org/abs/2501.08313.

Chien Van Nguyen, Ruiyi Zhang, Hanieh Deilamsalehy, Puneet Mathur, Viet Dac Lai, Haoliang Wang, Jayakumar Subramanian, Ryan A. Rossi, Trung Bui, Nikos Vlassis, Franck Dernoncourt, and Thien Huu Nguyen. Lizard: An efficient linearization framework for large language models, 2025. URL https://arxiv.org/abs/2507.09025.

Daniele Paliotta, Junxiong Wang, Matteo Pagliardini, Kevin Y. Li, Aviv Bick, J. Zico Kolter, Albert Gu, François Fleuret, and Tri Dao. Thinking slow, fast: Scaling inference compute with distilled reasoners, 2025. URL https://arxiv.org/abs/2502.20339.

Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Teddy Ferdinan, Haowen Hou, Przemysław Kazienko, et al. Eagle and finch: Rwkv with matrix-valued states and dynamic recurrence. *arXiv preprint arXiv:2404.05892*, 3, 2024.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A. Smith, and Lingpeng Kong. In *Proceedings of ICLR*, 2021.

Hao Peng, Jungo Kasai, Nikolaos Pappas, Dani Yogatama, Zhaofeng Wu, Lingpeng Kong, Roy Schwartz, and Noah A. Smith. ABC: Attention with bounded-memory control. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7469–7483, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.515. URL https://aclanthology.org/2022.acl-long.515/.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. HGRN2: Gated Linear RNNs with State Expansion. In *Proceedings of CoLM*, 2024.

Liliang Ren, Yang Liu, Yadong Lu, Yelong Shen, Chen Liang, and Weizhu Chen. Samba: Simple hybrid state space models for efficient unlimited context language modeling, 2025. URL https://arxiv.org/abs/2406.07522.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear Transformers Are Secretly Fast Weight Programmers. In *Proceedings of ICML*, 2021.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. Deltaproduct: Improving state-tracking in linear rnns via householder products, 2025. URL https://arxiv.org/abs/2502.10297.

Dustin Wang, Rui-Jie Zhu, Steven Abreu, Yong Shan, Taylor Kergan, Yuqi Pan, Yuhong Chou, Zheng Li, Ge Zhang, Wenhao Huang, and Jason Eshraghian. A systematic analysis of hybrid linear attention, 2025a. URL https://arxiv.org/abs/2507.06457.

Junxiong Wang, Daniele Paliotta, Avner May, Alexander M Rush, and Tri Dao. The mamba in the llama: Distilling and accelerating hybrid models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=uAzhODjALU.

Junxiong Wang, Wen-Ding Li, Daniele Paliotta, Daniel Ritter, Alexander M. Rush, and Tri Dao. M1: Towards scalable test-time compute with mamba reasoning models, 2025b. URL https://arxiv.org/abs/2504.10449.

Kaiyue Wen, Xingyu Dang, and Kaifeng Lyu. RNNs are not transformers (yet): The key bottleneck on in-context retrieval. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=h3wbI8Uk1Z.

Mingyu Yang, Mehdi Rezagholizadeh, Guihong Li, Vikram Appia, and Emad Barsoum. Zebra-llama: Towards extremely efficient hybrid models, 2025. URL https://arxiv.org/abs/2505.17272.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. *arXiv preprint arXiv:2312.06635*, 2023.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. *arXiv preprint arXiv:2412.06464*, 2024a.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *Proceedings of NeurIPS*, 2024b.

Lin Yueyu, Li Zhiyuan, Peter Yue, and Liu Xiao. Arwkv: Pretrain is not what we need, an rnn-attention-based language model born from transformer. *arXiv preprint arXiv:2501.15570*, 2025.

Luca Zancato, Arjun Seshadri, Yonatan Dukler, Aditya Golatkar, Yantao Shen, Benjamin Bowman, Matthew Trager, Alessandro Achille, and Stefano Soatto. B'mojo: Hybrid state space realizations of foundation models with eidetic and fading memory, 2024. URL https://arxiv.org/abs/2407.06324.

Michael Zhang, Kush Bhatia, Hermann Kumbong, and Christopher Ré. The Hedgehog & the Porcupine: Expressive Linear Attentions with Softmax Mimicry, 2024a. _eprint: arXiv:2402.04347.

Michael Zhang, Simran Arora, Rahul Chalamala, Benjamin Frederick Spector, Alan Wu, Krithik Ramesh, Aaryan Singhal, and Christopher Re. LoLCATs: On low-rank linearizing of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8VtGeyJyx9.

Yu Zhang, Songlin Yang, Rui-Jie Zhu, Yue Zhang, Leyang Cui, Yiqiao Wang, Bolun Wang, Freda Shi, Bailin Wang, Wei Bi, Peng Zhou, and Guohong Fu. Gated slot attention for efficient linear-time sequence modeling. In *NeurIPS*, 2024b. URL http://papers.nips.cc/paper_files/paper/2024/hash/d3f39e51f5f634fb16cc3e658f8512b9-Abstract-Conference.html.

Jingwei Zuo, Maksim Velikanov, Ilyas Chahed, Younes Belkada, Dhia Eddine Rhayem, Guillaume Kunsch, Hakim Hacid, Hamza Yous, Brahim Farhat, Ibrahim Khadraoui, Mugariya Farooq, Giulia Campesan, Ruxandra Cojocaru, Yasser Djilali, Shi Hu, Iheb Chaabane, Puneesh Khanna, Mohamed El Amine Seddik, Ngoc Dung Huynh, Phuc Le Khac, Leen AlQadi, Billel Mokeddem, Mohamed Chami, Abdalgader Abubaker, Mikhail Lubinets, Kacper Piskorski, and Slim Frikha. Falcon-h1: A family of hybrid-head language models redefining efficiency and performance, 2025. URL https://arxiv.org/abs/2507.22448.

A COMPLETE RESULTS ON RECALL-INTENSIVE BENCHMARKS

Tag	Selector	Signal / One-Line Procedure
UNIFORM	Uniform Interleave	Selects layers by evenly interleaving softmax layers at the target ratio.
Task-Guided Search (Heuristic-Based)		
KV	KV Retrieval	Importance from performance drop on a synthetic key-value dictionary lookup task when a layer is bypassed.
AR	Associative Recall	Importance from performance drop on a task to sum the values of prompted keys when a layer is bypassed.
AR-MH	Assoc. Recall—Multi-hop	As above, but with alias chains requiring multi-hop reasoning; performance drop defines importance.
VT	Variable Tracking	Importance from exact-set accuracy drop on a pointer-chasing task over shuffled assignments.
CWE	Common Words Extraction	Importance from set-match accuracy drop on a task to identify the K most frequent words in a long text.
ACT-MSE	Activation MSE	Mean-squared error on generic text between the final hidden states of a baseline vs. layer-bypassed model.
LM-PPL	LM Perplexity	Measures the increase in perplexity on a held-out corpus when a layer is bypassed.
Greedy Structural Search (Learning-Based)		
GR–S1	Greedy Removal (S1)	Starts with all softmax; greedily converts the layer to linear that hurts performance least after brief Stage-1 adaptation.
GR–S2	Greedy Removal (S2)	As above, but using a brief Stage-2 knowledge distillation for adaptation at each step.
GA–S1	Greedy Addition (S1)	Starts with all linear; greedily converts the layer to softmax that helps performance most after brief Stage-1 adaptation.
GA–S2	Greedy Addition (S2)	As above, but using a brief Stage-2 knowledge distillation for adaptation at each step.
AVG–S1	Rank-Avg Greedy (S1)	Averages the layer importance rankings from GR–S1 and GA–S1 before selecting the top-K layers.
AVG–S2	Rank-Avg Greedy (S2)	Averages the layer importance rankings from GR–S2 and GA–S2 before selecting the top-K layers.

Table 3: Layer-selection baselines and the tags used in figures. Layer bypass means applying an identity residual connection across the block's mixing sublayer.

	Llama-3.2-3B-Instruct				Qwen2.5-3B-Instruct			
Selector	**12.5%**	**25%**	**33%**	**50%**	**12.5%**	**25%**	**33%**	**50%**
Heuristic-Based								
UNIFORM	0.4134	0.4359	0.5477	0.6940	0.3718	0.6663	0.5927	0.8048
KV	0.2029	0.6051	0.6626	0.7538	0.2543	0.7539	0.7552	0.8257
AR	0.3229	0.5671	0.6948	0.8303	0.5267	0.6203	0.6685	0.8753
VT	0.1839	0.2012	0.4334	0.7538	0.2922	0.4780	0.5359	0.7409
CWE	0.3129	0.3579	0.6752	0.8394	0.2900	0.4907	0.7065	0.8444
ACT-MSE	0.2802	0.4534	0.5257	0.5580	0.3685	0.3658	0.5515	0.6725
LM-PPL	0.3672	0.4432	0.4692	0.6890	0.3964	0.4712	0.6646	0.6617
AR-MH	0.4044	0.5123	0.6219	0.8039	0.4364	0.7187	0.7217	0.8045
Learning-Based (S1 - MSE)								
GR-S1	0.2903	0.4508	0.5214	0.6435	0.3563	0.4827	0.6743	0.8209
GA-S1	0.3092	0.4193	0.4892	0.6569	0.3843	0.5408	0.6657	0.7873
AVG-S1	0.3108	0.4233	0.5355	0.6390	0.3960	0.4933	0.6441	0.8226
Learning-Based (S2 - KL)								
GR-S2	0.3084	0.4950	0.6991	0.7662	0.5804	0.8259	0.8541	0.8869
GA-S2	0.5389	0.6174	0.7003	0.7712	0.6946	0.8713	0.8743	0.9074
AVG-S2	0.4764	0.5580	0.6786	0.8111	0.7075	0.8205	0.8704	0.9051

Table 4: RULER performance for various layer selection strategies across different softmax ratios, for GDN-based hybrid students . The all-linear (0%) baselines are 0.0427 for Llama-3.2 and 0.1236 for Qwen2.5. The all-softmax teacher scores are 0.8934 and 0.9174, respectively.

	Llama-3.2-3B-Instruct				Qwen2.5-3B-Instruct			
Selector	**12.5%**	**25%**	**33%**	**50%**	**12.5%**	**25%**	**33%**	**50%**
Heuristic-Based								
UNIFORM	0.3013	0.2931	0.3947	0.6379	0.2686	0.6869	0.3303	0.7350
KV	0.2069	0.6461	0.6760	0.6942	0.1370	0.6788	0.6261	0.7096
AR	0.3820	0.5653	0.6860	0.7042	0.4746	0.5336	0.6688	0.7387
VT	0.1978	0.3385	0.3648	0.6960	0.1588	0.4183	0.4809	0.6279
CWE	0.3149	0.3258	0.6207	0.6779	0.0789	0.2087	0.5345	0.6842
ACT-MSE	0.2178	0.4537	0.5263	0.5672	0.1833	0.2377	0.3485	0.5889
LM-PPL	0.2922	0.4510	0.4982	0.7132	0.2423	0.2495	0.4773	0.5481
AR-MH	0.3539	0.4147	0.5472	0.6216	0.1407	0.6425	0.6434	0.7278
Learning-Based (S1 - MSE)								
GR-S1	0.2015	0.3548	0.5644	0.6007	0.2677	0.4465	0.5100	0.6697
GA-S1	0.2105	0.4365	0.4746	0.5563	0.2532	0.4247	0.5163	0.6234
AVG-S1	0.1951	0.4074	0.4628	0.6443	0.2414	0.4165	0.5227	0.6751
Learning-Based (S2 - KL)								
GR-S2	0.3303	0.5054	0.6933	0.6633	0.3612	0.6860	0.7459	0.7468
GA-S2	0.7060	0.7033	0.7114	0.7577	0.6180	0.7704	0.6878	0.8067
AVG-S2	0.6588	0.6806	0.7241	0.7060	0.5880	0.7532	0.7196	0.7641

Table 5: FDA performance for various layer selection strategies across different softmax ratios, for GDN-based hybrid students.

	Llama-3.2-3B-Instruct				Qwen2.5-3B-Instruct			
Selector	**12.5%**	**25%**	**33%**	**50%**	**12.5%**	**25%**	**33%**	**50%**
Heuristic-Based								
UNIFORM	0.7516	0.7525	0.8227	0.8452	0.7075	0.8515	0.8236	0.8776
KV	0.4671	0.7894	0.8110	0.8515	0.5311	0.8074	0.8101	0.8272
AR	0.6490	0.8191	0.8299	0.8542	0.7354	0.7759	0.7876	0.8866
VT	0.4761	0.6688	0.6895	0.8569	0.5572	0.7255	0.7507	0.8587
CWE	0.5878	0.6598	0.8290	0.8569	0.5302	0.7192	0.8020	0.8956
ACT-MSE	0.5896	0.7687	0.8101	0.8227	0.6049	0.7057	0.7930	0.8533
LM-PPL	0.6931	0.7525	0.7696	0.8362	0.6571	0.7453	0.8218	0.8353
AR-MH	0.7507	0.8128	0.8335	0.8461	0.6436	0.7948	0.7957	0.8200
Learning-Based (S1 - MSE)								
GR-S1	0.5779	0.6958	0.7480	0.8254	0.6688	0.7831	0.8326	0.8821
GA-S1	0.5707	0.7282	0.8146	0.8344	0.6553	0.8047	0.8569	0.8668
AVG-S1	0.5671	0.7192	0.7957	0.8254	0.6670	0.7975	0.8506	0.8866
Learning-Based (S2 - KL)								
GR-S2	0.6301	0.8110	0.8245	0.8425	0.8299	0.8875	0.8749	0.8929
GA-S2	0.8101	0.8263	0.8614	0.8605	0.8434	0.8812	0.8893	0.8875
AVG-S2	0.7885	0.8137	0.8565	0.8704	0.8128	0.8848	0.9001	0.9109

Table 6: SWDE performance for various layer selection strategies across different softmax ratios, for GDN-based hybrid students.

	Llama-3.2-3B-Instruct				Qwen2.5-3B-Instruct			
Selector	**12.5%**	**25%**	**33%**	**50%**	**12.5%**	**25%**	**33%**	**50%**
Heuristic-Based								
UNIFORM	19.1708	21.9026	23.8641	24.3945	7.5400	9.6984	9.0306	14.0742
KV	17.4030	25.5568	26.3946	29.5483	6.6478	10.8318	16.4796	15.2550
AR	18.2412	25.2227	27.8562	30.5521	8.7855	7.8277	9.8152	6.5062
VT	19.0819	24.3118	23.9263	29.9387	7.1499	8.7797	14.3150	18.8876
CWE	23.7679	23.2527	28.0014	30.3961	6.7367	12.9678	9.9249	7.2352
ACT-MSE	16.1512	22.0928	23.3075	25.4255	7.4720	5.3176	12.0061	9.6091
LM-PPL	18.5295	21.5863	22.0008	28.8905	9.0530	8.1341	7.8841	7.6171
AR-MH	21.8859	25.3047	26.6214	30.3687	9.8987	8.3828	12.2048	13.7225
Learning-Based (S1 - MSE)								
GR-S1	13.3918	20.7552	23.2197	27.3407	7.8245	7.0497	9.4220	8.6667
GA-S1	13.6481	17.8867	22.6633	29.2390	8.9412	9.0555	11.1751	9.1234
AVG-S1	15.0889	18.4342	24.3658	28.2178	7.6409	10.6217	10.1589	10.3181
Learning-Based (S2 - KL)								
GR-S2	18.0648	25.7848	30.4299	30.5907	12.1582	6.4855	7.8482	6.9539
GA-S2	25.9975	29.6941	30.8139	32.4805	11.4124	9.7799	12.0140	10.0936
AVG-S2	23.5556	29.2189	31.1063	32.1499	10.6181	6.4121	6.5623	11.3837

Table 7: SQuADv2 (F1) performance for various layer selection strategies across different softmax ratios, for GDN-based hybrid students.

B Elaboration on Early Stopping for Efficient Selection

Protocol. We study the sample efficiency of our one-swap selector (§3.2) at a fixed hybrid ratio of 1:3 (K=9 for Qwen2.5-3B-Instruct; K=7 for Llama-3.2-3B-Instruct). During Stage-2 we train for $4{,}550$ steps and, every 50 steps, compute the current top-K set of layers (from the one-swap importance scores). This yields 91 snapshot sets per model. To quantify stability we analyze each *rolling window* of the last R=10 snapshots using two complementary views:

- **Rolling pairwise similarity:** the mean pairwise Jaccard over the R sets.
- **Rolling backbone size:** the size of the intersection across the R sets (how many positions are "locked in").

We also relate snapshots to the final selection by reporting the fraction that are *within one swap* of the final consensus (Jaccard $\geq \frac{K-1}{K+1}$; i.e., 0.80 for K=9 and 0.75 for K=7).[3]

Reliable selections emerge well before 4550 steps. Two patterns are consistent across both teachers:

- **Qwen2.5-3B-Instruct (K=9).** The run-best set first appears by step 850. From step 1500 onward, 95% of snapshot sets are within one swap of the final consensus; the 10-snapshot rolling Jaccard is high on average (≈ 0.95), and rises to 0.99 beyond step 2350.

  By step 1900, the last R snapshots share an 8/9 backbone with at most two candidates for the remaining slot; any one-swap variant at this point attains RULER within 0.007–0.009 absolute points of the run-best (0.8662 vs. $0.8592/0.8582/0.8574$).
- **Llama-3.2-3B-Instruct (K=7).** A 6/7 backbone appears by step 750 (mean window Jaccard ≈ 0.91). The near-optimal set that differs by a single layer first appears at step 1200; from step 1200 onward, 100% of snapshots are within one swap of the final consensus. Stopping here gives RULER 0.6971, within 0.004 absolute of the run-best 0.7011 and comparable to the best late-appearing set.

These observations (i) The selector's rankings stabilize far earlier than the full 4500-step budget; (ii) once the windowed sets agree on $K-1$ layers, the remaining degree of freedom is small and can be resolved cheaply; (iii) one-swap neighbors of the eventual best set typically match downstream RULER within 0.1–1.0 absolute points, so stopping once the $K-1$ backbone is stable is a sound efficiency–quality trade-off.

A conservative choice (see rule below) would have stopped at ~ 1900 steps for Qwen and ~ 1200 steps for Llama—consuming 42% and 27% of the 4550-step budget, respectively (i.e., 58–74% fewer tokens for the selection pass).

Practical recipe (rolling-Jaccard early stop). Let S_t be the top-K set at step t and $W_t = \{S_{t-9}, \ldots, S_t\}$. Define

$$\text{Backbone}_t = \bigcap_{S \in W_t} S, \quad \text{JaccardMean}_t = \frac{2}{R(R-1)} \sum_{i<j} \text{Jac}(S_i, S_j).$$

Stop at the first step t satisfying:

1. $\text{JaccardMean}_t \geq 0.90$,
2. $|\text{Backbone}_t| \geq K - 1$, and
3. $|\bigcup_{S \in W_t} S| \leq K + 1$ (at most two options for the remaining slot).

(Optional) Stop when (3) first becomes true and $S_t \neq S_{t-1}$ to pick the newer of the two candidates.

[3] For fixed set size K, replacing one layer yields intersection $K-1$ and union $K+1$, hence Jaccard $(K-1)/(K+1)$.

C Complete Layer Importance Rankings

For all methods that produce a scalar importance score per layer, we obtain hybrid architectures at target softmax ratios (12.5%, 25%, 33%, 50%) by taking the top-K most important layers according to that ranking (with K determined by the ratio and total depth L). In this section we report the *full* importance ranking for each such method. Layer indices are zero-based. Methods such as PostNAS and Smart do not provide layerwise importance scores, so they are omitted here.

C.1 Qwen2.5-3B-Instruct

Selector	Layer indices (most → least important)
KV	[1, 0, 26, 19, 18, 20, 5, 17, 27, 6, 15, 22, 24, 16, 3, 11, 23, 21, 28, 8, 14, 25, 2, 29, 32, 12, 13, 9, 4, 10, 31, 34, 35, 30, 33, 7]
AR	[0, 1, 27, 18, 20, 25, 24, 26, 21, 8, 12, 19, 23, 7, 35, 17, 33, 22, 28, 16, 32, 30, 34, 9, 29, 2, 6, 5, 31, 4, 13, 10, 14, 15, 3, 11]
VT	[0, 1, 19, 26, 28, 25, 35, 10, 15, 17, 3, 7, 27, 29, 16, 14, 30, 34, 32, 31, 23, 33, 9, 13, 18, 8, 2, 21, 11, 12, 22, 24, 20, 5, 4, 6]
CWE	[0, 1, 22, 24, 16, 13, 26, 2, 27, 19, 20, 11, 23, 6, 31, 28, 29, 33, 4, 8, 34, 7, 30, 32, 9, 25, 3, 5, 21, 15, 17, 18, 35, 10, 14, 12]
Act-MSE	[0, 1, 35, 34, 31, 33, 32, 30, 8, 12, 27, 3, 4, 2, 6, 5, 28, 10, 9, 29, 11, 7, 14, 13, 26, 25, 16, 15, 18, 24, 17, 23, 20, 19, 22, 21]
LM-PPL	[0, 1, 35, 34, 32, 31, 33, 30, 27, 12, 6, 5, 9, 8, 29, 2, 4, 7, 10, 11, 25, 28, 16, 14, 13, 26, 24, 20, 3, 22, 23, 15, 18, 21, 19, 17]
AR-MH	[0, 1, 27, 21, 26, 16, 20, 5, 23, 24, 18, 6, 13, 3, 9, 22, 8, 17, 33, 35, 19, 4, 25, 12, 30, 7, 29, 34, 14, 15, 10, 2, 28, 11, 32, 31]
GA-S1	[33, 32, 34, 31, 35, 28, 29, 27, 21, 22, 19, 30, 24, 16, 23, 26, 12, 17, 18, 20, 14, 25, 10, 3, 11, 6, 13, 7, 9, 15, 0, 4, 8, 2, 5, 1]
GR-S1	[33, 32, 34, 35, 31, 27, 28, 30, 21, 29, 22, 19, 26, 25, 16, 24, 23, 17, 18, 14, 15, 12, 20, 13, 11, 10, 8, 9, 7, 6, 3, 5, 4, 0, 2, 1]
AVG-S1	[33, 32, 34, 31, 35, 28, 27, 29, 21, 30, 22, 19, 16, 24, 26, 23, 17, 25, 18, 12, 14, 20, 10, 11, 13, 15, 3, 6, 7, 9, 8, 0, 4, 5, 2, 1]
GA-S2 (Ours)	[20, 32, 33, 21, 22, 25, 17, 19, 5, 31, 4, 3, 10, 30, 26, 29, 27, 13, 0, 28, 15, 23, 6, 12, 24, 7, 18, 9, 34, 14, 11, 8, 16, 35, 2, 1]
GR-S2	[21, 33, 19, 27, 0, 32, 17, 22, 20, 25, 23, 18, 24, 15, 29, 12, 26, 31, 16, 3, 10, 13, 14, 28, 30, 5, 7, 8, 11, 4, 35, 6, 9, 2, 34, 1]
AVG-S2	[21, 33, 32, 20, 19, 22, 17, 25, 27, 0, 31, 29, 3, 26, 23, 10, 5, 15, 24, 18, 30, 12, 13, 4, 28, 16, 7, 14, 6, 8, 11, 9, 34, 35, 2, 1]

Table 8: Complete layer-importance rankings for `Qwen2.5-3B-Instruct`. Each row lists all $L = 36$ layers from most to least important.

C.2 Llama-3.2-3B-Instruct

Selector	Layer indices (most → least important)
KV	[0, 7, 5, 4, 8, 11, 14, 2, 1, 3, 6, 23, 10, 20, 26, 17, 22, 9, 24, 21, 25, 18, 16, 19, 12, 13, 27, 15]
AR	[0, 16, 11, 14, 7, 5, 9, 13, 2, 12, 1, 8, 27, 26, 10, 6, 24, 15, 3, 20, 18, 19, 17, 21, 25, 4, 23, 22]
VT	[0, 5, 4, 11, 3, 12, 10, 1, 2, 17, 9, 13, 15, 16, 18, 23, 8, 14, 21, 24, 20, 25, 26, 22, 6, 27, 19, 7]
CWE	[0, 5, 12, 8, 9, 4, 1, 13, 14, 10, 21, 24, 16, 22, 15, 27, 25, 20, 6, 2, 26, 23, 18, 3, 11, 17, 19, 7]
ACT-MSE	[0, 1, 27, 24, 25, 2, 26, 4, 15, 23, 19, 21, 3, 18, 20, 14, 16, 5, 22, 17, 13, 6, 7, 12, 11, 10, 8, 9]
LM-PPL	[0, 1, 27, 2, 24, 3, 26, 25, 4, 14, 15, 19, 16, 5, 20, 23, 12, 17, 10, 21, 13, 18, 6, 22, 9, 11, 7, 8]
AR-MH	[0, 13, 12, 16, 11, 7, 23, 14, 10, 5, 21, 25, 9, 8, 19, 17, 2, 6, 4, 3, 1, 26, 18, 24, 15, 22, 27, 20]
GA-S1	[26, 27, 25, 24, 13, 20, 23, 7, 10, 22, 9, 12, 19, 8, 14, 15, 21, 11, 16, 17, 18, 2, 5, 6, 4, 1, 0, 3]
GR-S1	[26, 27, 24, 25, 23, 12, 22, 13, 14, 21, 10, 19, 11, 15, 9, 20, 8, 7, 16, 18, 17, 6, 5, 4, 3, 2, 1, 0]
AVG-S1	[26, 27, 24, 25, 23, 13, 22, 12, 10, 20, 14, 19, 7, 9, 21, 15, 8, 11, 16, 17, 18, 5, 6, 2, 4, 1, 3, 0]
GA-S2 (OURS)	[14, 8, 5, 12, 15, 13, 2, 26, 24, 16, 17, 18, 21, 10, 25, 20, 19, 23, 22, 27, 9, 7, 6, 4, 1, 0, 11, 3]
GR-S2	[0, 1, 12, 2, 13, 5, 10, 14, 8, 7, 9, 6, 3, 11, 26, 15, 16, 4, 22, 24, 27, 25, 19, 17, 18, 23, 21, 20]
AVG-S2	[12, 5, 14, 2, 8, 13, 10, 15, 26, 0, 1, 16, 24, 7, 9, 6, 17, 18, 25, 22, 19, 21, 3, 11, 27, 4, 20, 23]

Table 9: Complete layer-importance rankings for `Llama-3.2-3B-Instruct`. Each row lists all $L = 28$ layers from most to least important.

D LAYER-SELECTION PATTERNS AND SPATIAL ORGANIZATION

We now examine where in depth the selected softmax layers tend to lie, and whether our selector prefers isolated layers or groups of consecutive layers.

Setup. For each teacher we take the GA–S2 ranking $\mathcal{R} = (\ell_1, \ldots, \ell_L)$ from Appendix C, ordered from most to least important. For a softmax budget K we define the selected set $S_K = \{\ell_1, \ldots, \ell_K\}$. To quantify how much the selected layers cluster in depth, we use the *adjacency index*

$$A_K = \big|\{\, i \in S_K : i+1 \in S_K \,\}\big|,$$

i.e., the number of pairs of consecutive layers that are both selected. For a uniformly random K-subset of $\{0, \ldots, L-1\}$, the expected value is $\mathbb{E}[A_K] \approx K(K-1)/L$, so values substantially above this baseline indicate more clustering than would be obtained by chance. Figure 7 shows the selected indices across budgets, and Figure 8 compares observed and expected adjacency counts.

Results and discussion. For **Qwen2.5-3B-Instruct** (L=36), GA–S2 produces selected sets that are visibly concentrated in a few depth ranges. At a 25% budget (K=9), we obtain $A_K = 4.0$ versus a random baseline of 2.0; at 33% (K=12), $A_K = 7.0$ versus 3.68; and at 50% (K=18), $A_K = 11.0$ versus 8.49. The plot in Figure 7 show that several of these adjacent pairs occur repeatedly around layers roughly 3–5, 19–22, and 31–33, while the remaining layers are used more sparsely. Thus, the selector does not simply spread the softmax layers uniformly but repeatedly reuses a small number of depth regions as the budget increases.

For **Llama-3.2-3B-Instruct** (L=28), the effect is weaker but still present. At 25% (K=7), $A_K = 3.0$ versus a baseline of 1.50; at 33% (K=9), $A_K = 3.0$ versus 2.58; and at 50% (K=14), $A_K = 6.0$ versus 6.50. The selected layers tend to form one main group in the middle of the network (around layers 12–18), with a smaller number of layers near the input and output.

Overall, both models show some degree of clustering beyond what would be expected from a random K-subset, but the pattern (multiple groups versus a single main group) depends on the teacher architecture.

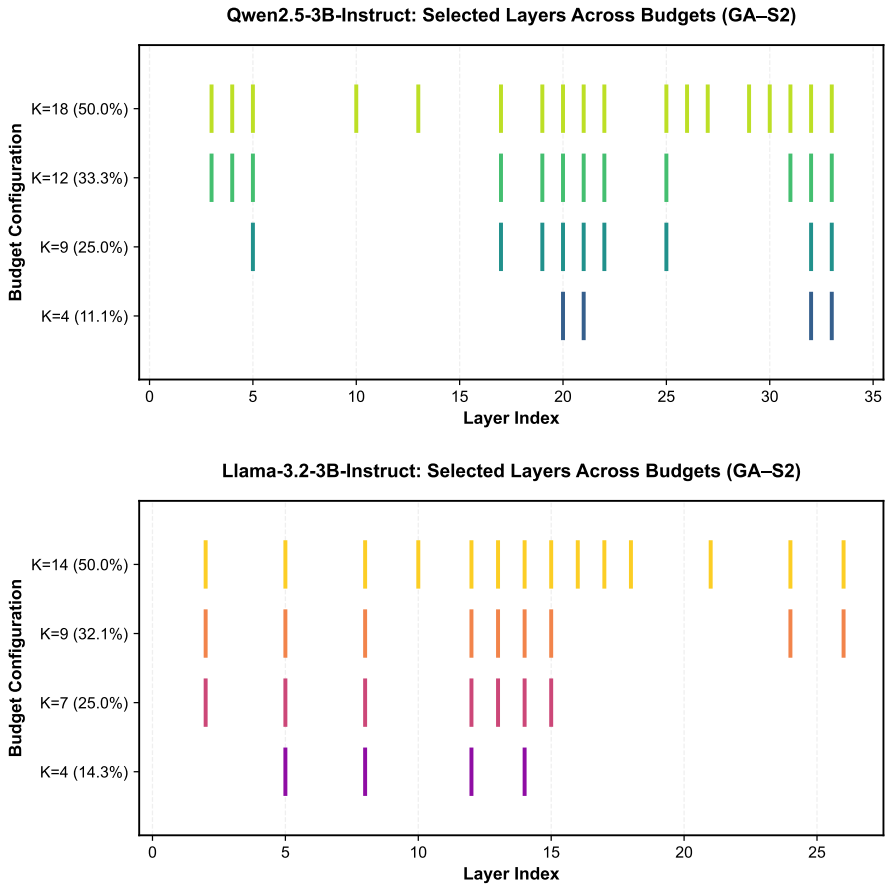


Figure 7: Visualization of selected layers for Qwen2.5-3B-Instruct (top) and Llama-3.2-3B-Instruct (bottom) across budgets (12.5%, 25%, 33%, 50%). Each vertical tick marks a selected layer index.

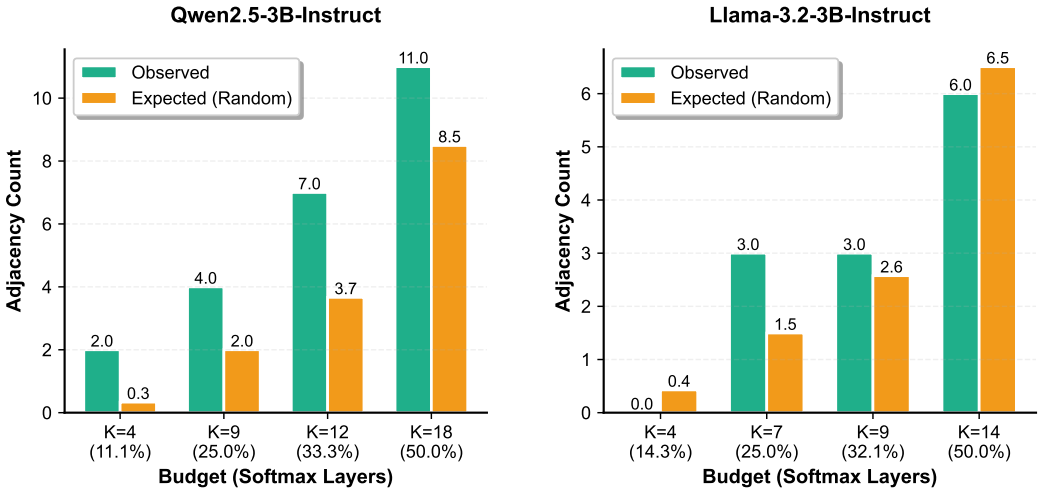


Figure 8: Observed (solid) vs. random-baseline expected (dashed) adjacency counts A_K for Qwen2.5-3B (left) and Llama-3.2-3B (right).

E Distance-Regularized Selection (Diversification Ablation)

To probe whether clustering is redundant, we evaluate a re-weighted greedy rule for selecting K layers:

$$\tilde{\mathcal{I}}(\ell \mid S) \;=\; \mathcal{I}(\ell) \;-\; \lambda \sum_{j \in S} \exp\!\left(-\frac{|\ell - j|}{\sigma}\right),$$

with $\lambda > 0$, $\sigma > 0$. Here S is the set of softmax layers selected so far and $\mathcal{I}(\ell)$ is the original GA–S2 importance score. The exponential term penalizes placing a new softmax layer too close (in depth) to previously selected ones, nudging the selector toward more spatially diverse configurations without discarding the model-intrinsic KL signal.

We instantiate this diversification for Qwen2.5-3B-Instruct with a GDN student at a fixed 25% softmax ratio (K=9), and sweep $\lambda \in \{0.025, 0.05\}$ and $\sigma \in \{1, 2\}$. All other training and evaluation settings are kept identical to the main GA–S2 runs.

λ	σ	RULER (4096)	**Selected layers**
0 (GA–S2)	–	0.8713	[20, 32, 33, 21, 22, 25, 17, 19, 5]
0.025	1	0.8509	[20, 32, 25, 17, 22, 5, 33, 10, 3]
0.025	2	0.8244	[20, 32, 25, 5, 17, 10, 33, 0, 22]
0.050	1	0.8334	[20, 32, 25, 17, 5, 10, 22, 0, 29]
0.050	2	0.8303	[20, 32, 5, 25, 10, 17, 0, 33, 13]

Table 10: Distance-regularized GA–S2 selection on Qwen2.5-3B-Instruct with a GDN student at a 25% softmax ratio. The λ=0 row corresponds to our default GA–S2 selector without regularization; the last column lists the resulting softmax layer indices.

As shown in Table 10, none of the distance-regularized variants outperform the unregularized GA–S2 selector. A mild penalty (λ=0.025, σ=1) yields a small degradation (0.8509 vs. 0.8713 on RULER), while stronger or more broadly supported penalties lead to larger drops. This suggests that the clustering observed in our selections is not merely redundant: forcing softmax layers to spread out in depth tends to remove genuinely useful local groupings. At the same time, the λ=0.025, σ=1 configuration may be acceptable when a slightly more uniform spatial allocation is desired and a modest recall loss (about two points on RULER) is tolerable.

F Extended Long-Context Evaluation via Needle-in-a-Haystack

In the main text, long-context behavior is evaluated primarily through RULER and SWDE (§4, §4.1), whose contexts are below 10k tokens, and our distillation pipeline (§3.1) is trained on generic text with comparatively shorter sequence lengths. This leaves open whether the distilled hybrid model recovers teacher-like retrieval ability at substantially longer sequences than those used during distillation and benchmark evaluation. To probe this, we perform an additional needle-in-a-haystack (NiHA) experiment.

We consider the Qwen2.5-3B-Instruct teacher and its corresponding hybrid student with a 25% softmax / 75% GDN configuration selected by our method. For each context length, we construct inputs by embedding a single target "needle" span into a long filler context and measure retrieval accuracy, defined as the fraction of cases where the model correctly identifies the target span. We evaluate across exponentially increasing context window sizes from 8k to 128k tokens. Results are reported in Table 11.

Context length (tokens)	Teacher	Hybrid student
8,192	1.000	1.000
16,384	1.000	0.998
32,768	1.000	0.998
65,536	1.000	0.994
131,072	0.954	0.684

Table 11: Needle-in-a-haystack retrieval accuracy as a function of context length for Qwen2.5-3B-Instruct (teacher) and the corresponding hybrid student (25% softmax, 75% GDN layers).

The hybrid model maintains near-perfect retrieval accuracy up to 65,536 tokens, closely tracking the teacher with only minor degradation. At 131,072 tokens both models begin to degrade, with a larger drop for the hybrid student. These results indicate that the proposed layer selection and distillation procedure successfully preserves long-context retrieval well beyond the context lengths used during distillation and primary benchmark evaluations, while leaving further improvements at extreme lengths as an interesting direction for future work.

G Additional Scaling Results for Qwen2.5 Teachers

To verify that our KL-guided layer selection method scales across model sizes within a family, we also distill GDN-based hybrid students from two additional Qwen2.5 teachers:

- **Qwen2.5-1.5B-Instruct**, with RULER score 0.8742.
- **Qwen2.5-7B-Instruct**, with RULER score 0.9445.

We use the same DCLM mixture and distillation pipeline as in the main Qwen2.5-3B experiments, and evaluate at 25% and 33% softmax ratios. As in the main text, we compare against Uniform, AR, AR-MH, Act-MSE, LM-PPL, and SMART. Our selector GA-S2 remains consistently stronger than all baselines, particularly in the low-budget regime.

Model / Ratio	Uniform	AR	AR-MH	Act-MSE	LM-PPL	SMART	**GA-S2**
Qwen2.5-1.5B-Instruct (teacher RULER: 0.8742)							
25%	0.4778	0.5096	0.4243	0.3807	0.4271	0.5098	**0.5408**
33%	0.5651	0.5552	0.5229	0.4374	0.5056	0.6479	**0.6953**
Qwen2.5-7B-Instruct (teacher RULER: 0.9445)							
25%	0.7357	0.7453	0.7322	0.6469	0.6544	0.8158	**0.8584**
33%	0.7516	0.8423	0.8533	0.7227	0.6590	0.8949	**0.9110**

Table 12: RULER performance of GDN-based hybrid students distilled from smaller (1.5B) and larger (7B) Qwen2.5 teachers at 25% and 33% softmax ratios. Our GA-S2 selector consistently outperforms all baselines across scales.