# HYBRID ARCHITECTURES FOR LANGUAGE MODELS: SYSTEMATIC ANALYSIS AND DESIGN INSIGHTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

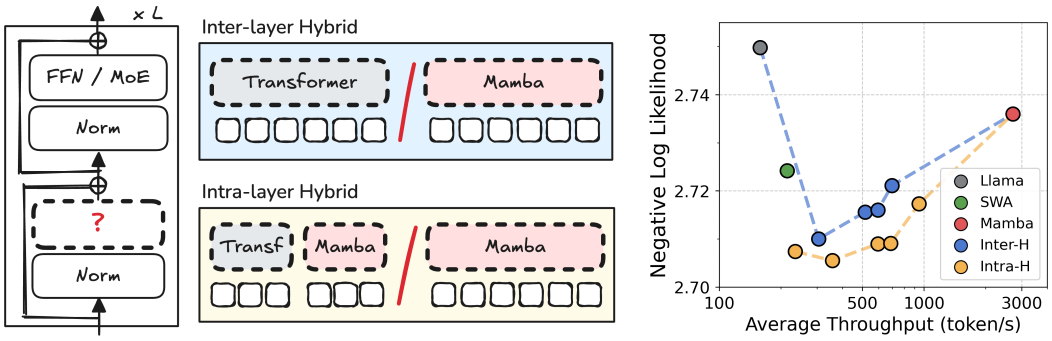
Recent progress in large language models demonstrates that hybrid architectures–combining self-attention mechanisms with structured state space models like Mamba–can achieve a compelling balance between modeling quality and computational efficiency, particularly for long-context tasks. While these hybrid models show promising performance, systematic comparisons of hybridization strategies and analyses on the key factors behind their effectiveness have not been clearly shared to the community. In this work, we present a holistic evaluation of hybrid architectures based on inter-layer (sequential) or intra-layer (parallel) fusion. We evaluate these designs from a variety of perspectives: language modeling performance, long-context capabilities, scaling analysis, and training and inference efficiency. By investigating the core characteristics of their computational primitive, we identify the most critical elements for each hybridization strategy and further propose optimal design recipes for both hybrid models. Our comprehensive analysis provides practical guidance and valuable insights for developing hybrid language models, facilitating the optimization of architectural configurations.

## 1 INTRODUCTION

Recent language models (Llama Team, 2024; Microsoft Research, 2024; DeepSeek-AI, 2024; OpenaAI et al., 2024; Qwen Team, 2025; Gemini Team, 2025) have demonstrated strong scalability and human-like performance across a wide range of tasks. Most of these models are based on the Transformer architecture (Vaswani et al., 2017), which alternates self-attention and feed-forward layers. However, the self-attention mechanism exhibits quadratic complexity with respect to input sequence length, leading to slow inference and a substantial memory footprint. To address these limitations, a new class of computational primitives—state space models (SSMs)—has emerged, inspired by signal processing (Gu et al., 2020; 2021a;b; 2022; Fu et al., 2022; Gu & Dao, 2023; Dao & Gu, 2024; Yang et al., 2024). These architectures scale more efficiently to long sequences by compressing prior context into a finite-dimensional state. Among these, the Mamba model (Gu & Dao, 2023; Dao & Gu, 2024) stands out for being competitive with Transformer on language modeling, while also accelerating training speed through work-efficient parallel scan algorithms (Blelloch, 1990; Smith et al., 2022).

These new primitives expand the architecture design space, opening up new possibilities for hybrid models that leverage the strengths of different architectural choices (Glorioso et al., 2024b; Ren et al., 2024; Lieber et al., 2024; Wang et al., 2024a; Poli et al., 2024; Dong et al., 2024; Ren et al., 2025; Basant et al., 2025). Notably, mixing Transformer and Mamba blocks often outperforms homogeneous architectures, while maintaining high efficiency. Most prior work on hybrid models has focused on the sequential interleaving of standard Transformer and Mamba blocks—an *inter*-layer hybrid approach (Glorioso et al., 2024b; Ren et al., 2024; Jamba Team et al., 2024; Hunyuan Team et al., 2025). This practical strategy allows for a balance between model quality and throughput by adjusting the ratio of quadratic (Transformer) to linear (Mamba) attention blocks. In addition, a few *intra*-layer hybrid models have been proposed, which fuse the two primitives in parallel within individual layers. These approaches use either head-wise (Dong et al., 2024; Xiao et al., 2025) or sequence-wise splits (Zhang et al., 2024; Li et al., 2025b) to further combine the benefits of both architectures at a finer granularity. Figure 1a summarizes which attention primitives each hybridization type can use.

Despite the emergence of various hybrid architectures, the current literature lacks openly shared, in-depth analysis of hybridization strategies, making it difficult to understand their relative strengths and




(a) Overview of hybrid architecture

(b) Pareto frontier of throughput

Figure 1: **(a) Two hybridization strategies construct attention using either Transformer (or intra-hybrid) or Mamba blocks.** For clarity, Transformer primitive denotes multi-head attention (or efficient variants). Varying ratios of these blocks controls the degree of hybridization. In intra-hybrid blocks, all heads are split into two halves, which are then processed by half-sized Transformer and Mamba blocks, respectively. **(b) The hybrid architectures achieve superior quality-throughput trade-offs compared to homogeneous architectures.** Negative log likelihood (i.e., loss) is measured on the DCLM validation set, and inference throughput is averaged across total lengths of 2K, 4K, 8K, 16K, and 32K, with the prompt length fixed at 512. All models have 1B parameters and are trained with the same FLOPs budget of 4.5e20 and 8K context length. For hybrids, we connect results for different block ratios (1:0, 1:1, 1:3, 1:5, 1:12, 0:1)—where each ratio denotes (Transformer / Intra-hhybrid : Mamba blocks)—with dashed lines. In the sliding window attention (SWA) model, global attention and SWA are interleaved at a 1:5 ratio, with a window size of 512 and an attention sink size of 64.

design trade-offs (Sun et al., 2025). In particular, most prior works focus on introducing specific hybrid architectures rather than providing detailed, systematic comparisons across possible hybridization approaches. As a result, the key intuitions driving final design choices, as well as the relative merits of different strategies from multiple perspectives, remain open questions for the community.

In this work, we address these research questions by conducting a holistic evaluation of hybrid architectures via comprehensive, multi-faceted comparisons. Specifically, for model architecture choices, we compare inter-layer and intra-layer hybridization strategies with Transformer (Llama Team, 2024), Mamba (Dao & Gu, 2024), and striped models of global and sliding window attention (Beltagy et al., 2020; Gemma Team et al., 2025) with attention sink (Xiao et al., 2023). From *quality* perspectives, we identify optimal design choices for hybrid models by providing key insights into critical architectural considerations, based on language modeling perplexity (§4.1) and extensive ablation studies (§4.5 and §4.6). Additionally, we analyze long-context retrieval performance (Rae et al., 2019a; Kamradt, 2023) with the characteristics of each computational primitive (§4.3). Our exploration of Mixture-of-Experts (Fedus et al., 2022; DeepSeek-AI, 2024) and compute-optimal scaling (Kaplan et al., 2020; Hoffmann et al., 2022) further offers practical guidance for scaling up hybrid models (§4.4). On the *efficiency* front, we conduct an in-depth comparison of both training and inference, measuring training time, memory usage, inference throughput, and cache size (§4.2). Based on our comprehensive experiments, we summarize the principal insights regarding hybrid architectures as follows:

**[Quality]** Both hybrid model types outperform homogeneous architectures up to 2.9% accuracy and even surpass widely adopted SWA models in quality. Intra-layer hybridization shows best pareto-frontier of model quality and efficiency (see Figure 1b and Table 2).

**[Quality]** While baselines show poor in-context retrieval and length generalization, all hybrid architectures achieve robust and superior long-context retrieval (see Figures 3c and 4).

**[Quality]** Hybrid models are fully compatible with MoE structures and achieve an optimal compute-scaling slope between those of Transformer and Mamba (see Table 3).

**[Efficiency]** By fully leveraging Mamba's efficiency strengths (i.e., linear complexity with respect to sequences), hybrid models achieve fast end-to-end training time (see Figure 2) and high inference throughput with lower cache sizes (see Figures 1b, 3a, and 3b).

**[Design]** Insights about optimal block ratios, ordering of computational primitives for hybrid architectural configurations are thoroughly explored (see Tables 4 and 5).

Table 1: **Overall comparison of per-block compute and memory costs across different primitives.** For simplicity, we exclude the additional term of 6 times the block parameter count from FLOPs. Here, $N$ denotes number, $d$ is dimension, and $L$ is sequence length. $L_{\text{swa}}$ is the sum of sliding window size and attention sink sizes, and $d_{\text{ssm}}$ is typically $2d_{\text{model}}$. For Transformers, we assume grouped-query attention, and cache sizes are calculated with `bfloat16` precision.

| | Compute | Memory | |
|---|---|---|---|
| **Models** | FLOPs per Sample | Parameter Counts | Total Cache Size |
| Llama | $12d_{\text{model}}L_{\text{ctx}} \times (L_{\text{ctx}}+1)/2$ | $2d_{\text{model}}d_{\text{model}}$ $+2d_{\text{model}}d_{\text{head}}N_{kv}$ | $2d_{\text{head}}N_{kv}L_{\text{ctx}}$ $+2d_{\text{head}}N_{kv}L_{\text{ctx}}$ |
| Sliding Window | $12d_{\text{model}}L_{\text{swa}}$ $\times((L_{\text{swa}}+1)/2+(L_{\text{ctx}}-L_{\text{swa}}))$ | $2d_{\text{model}}d_{\text{model}}$ $+2d_{\text{model}}d_{\text{head}}N_{kv}$ | $2d_{\text{head}}N_{kv}L_{\text{swa}}$ $+2d_{\text{head}}N_{kv}L_{\text{swa}}$ |
| Mamba | $3L_{\text{ctx}} \times (9d_{\text{ssm}}d_{\text{state}}+2d_{\text{ssm}})$ | $d_{\text{model}}(2d_{\text{ssm}}+2d_{\text{state}}+N_{\text{head}})$ $+d_{\text{state}}(N_{\text{conv}}+d_{\text{model}})+2N_{\text{head}}$ | $2d_{\text{ssm}}d_{\text{state}}$ $+2N_{\text{conv}}(2d_{\text{state}}+d_{\text{ssm}})$ |

## 2 Background

**Transformer architecture.** The Transformer architecture (Vaswani et al., 2017) consists of two main components: the multi-head attention (MHA) module and a feed-forward network (FFN). The MHA mechanism leverages multiple attention heads to capture diverse dependencies within the input sequence. For each head, attention is computed as follows:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \texttt{softmax}\left(\mathbf{Q}\mathbf{K}^T/\sqrt{d_k}\right)\mathbf{V},$$

where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are derived from learned linear projections of the input using $\mathbf{W}_\ell^Q$, $\mathbf{W}_\ell^K$, and $\mathbf{W}_\ell^V$, respectively. The outputs of all heads are concatenated and transformed by $\mathbf{W}_\ell^O$ to restore the original model dimension. To reduce key-value cache size, recent models divide query heads into groups, with each group sharing a single key and value head—a structure known as grouped-query or multi-query attention (Ainslie et al., 2023). On the other hand, sliding window attention (SWA) (Beltagy et al., 2020; Jiang et al., 2023; Gemma Team et al., 2025; Cohere et al., 2025; OpenAI et al., 2025) reduces the context length over which queries attend to key and value states. For the FFN component, recent models adopt a gating structure based on the SiGLU mechanism, which is a variant of SwiGLU (Shazeer, 2020; Chowdhery et al., 2023) but uses the SiLU activation instead of Swish. This FFN can be replaced with a Mixture-of-Experts layer (Shazeer et al., 2017; Fedus et al., 2022), where multiple FFNs (experts) are adaptively routed for each token.

**Mamba architecture.** The Mamba architecture (Gu & Dao, 2023) is a recent advancement in sequence modeling, building on structured state space models like S4 (Gu et al., 2021a), H3 (Fu et al., 2022), and S4D (Gu et al., 2022). Mamba replaces the attention mechanism with a state space model (SSM) layer, enabling efficient and expressive modeling of long-range dependencies:

$$\mathbf{h}_t = \bar{\mathbf{A}}\mathbf{h}_{t-1} + \bar{\mathbf{B}}\mathbf{x}_t, \quad \mathbf{y}_t = \mathbf{C}\mathbf{h}_t + \mathbf{D}\mathbf{x}_t$$

Here, $\mathbf{h}_t$ represents the latent state at time $t$, $\mathbf{x}_t$ is the input, and $\mathbf{y}_t$ is the output. In practice, $\bar{\mathbf{A}}$ is discretized using zero-order hold (ZOH), and $\bar{\mathbf{B}}$ is discretized via the Euler method. A key innovation is the input-dependent modulation of the state space parameters $\mathbf{B}$, $\mathbf{C}$, and the discretization step $\mathbf{\Delta}$, allowing the model to flexibly control context retention and input integration for each token. This selectivity enables Mamba to match the modeling quality of Transformers. Additionally, inspired by gated attention units (Hua et al., 2022), Mamba incorporates a SiLU-based gating mechanism. Mamba 2 (Dao & Gu, 2024) advances this design by introducing the state space duality (SSD) layer, which reformulates SSM computations as matrix multiplications highly optimized for modern hardware (e.g., GPUs, TPUs). This results in significantly faster training and improved practical efficiency. Throughout this paper, we primarily utilize the Mamba 2 architecture as the computational primitive, with each block always followed by a FFN layer, which can optionally be replaced with a MoE layer.

**Compute and memory costs comparison.** When analyzing costs in hybrid architectures (also for SWA models combining global and local attention), we reference the per-block costs in Table 1 (see Appendix D for details). For example, in a 1B model with 8K context, Transformer uses about 18% more FLOPs per sample than Mamba, due to its quadratic scaling versus Mamba's linear scaling. From a memory perspective, Mamba uses 2.5× more parameters per block (25M vs. 10M), but its cache size is 95% smaller than that of a Transformer (256 MiB vs. 13.4 MiB).

# 3 HYBRID ARCHITECTURE

Hybrid architectures can be categorized by their integration approach: inter-layer and intra-layer hybridization. We define each strategy and outline related research questions below.

## 3.1 INTER-LAYER HYBRID MODEL

**Definition.** The inter-layer hybrid model alternates softmax attention layers with linear sequence modeling layers at specific intervals (see Figure 1a). A key design choice in this approach is determining the order and proportion in which Transformer and Mamba primitives are arranged and interpolated. Its practical effectiveness and straightforward implementation have made the inter-layer approach a dominant strategy in hybrid architecture development.

**Related works.** Most hybrid models combine Mamba (Gu & Dao, 2023; Dao & Gu, 2024) with various attention mechanisms to enhance quality and efficiency. Notable examples—H3 (Fu et al., 2022), MambaFormer (Park et al., 2024), Zamba (Glorioso et al., 2024b;a), Jamba (Lieber et al., 2024; Jamba Team et al., 2024), Samba (Ren et al., 2024; 2025), Hunyuan-TurboS (Hunyuan Team et al., 2025), Nemotron nano 2 (Basant et al., 2025), and IBM Granite 4.0—leverage Mamba's efficiency with global or local attention, scaling to over 1M context length. Some post-training approaches—such as MambaInLLaMA (Wang et al., 2024a), MOHAWK (Bick et al., 2024), Zebra-Llama (Yang et al., 2025), and Jet-Nemotron (Gu et al., 2025)—use knowledge distillation to convert parts of a pretrained Transformer into linear modules, while STAR (Thomas et al., 2024) and Composer (Acun et al., 2025) perform architecture search. Other hybrids—RecurrentGemma (Botev et al., 2024), Griffin (De et al., 2024), Titans (Behrouz et al., 2024), RWKV-X (Hou et al., 2025), MiniMax-01 (Li et al., 2025a), Qwen3-Next, and Kimi Linear (Kimi Team, 2025)—combine diverse self-attention variants (Beltagy et al., 2020; DeepSeek-AI, 2024; Qin et al., 2024; Lu et al., 2025; Qiu et al., 2025) with different linear modules (Qin et al., 2022; De et al., 2024; Yang et al., 2024; Peng et al., 2025).

**Block ratios in inter-layer hybrid.** One of the key research questions in designing inter-layer hybrids is determining the optimal ratio of Transformer blocks to Mamba blocks when stacking layers. This ratio determines the degree of interpolation between two computational primitives, impacting not only model quality but also efficiency metrics like throughput and cache size. Prior work (Jamba Team et al., 2024; Lieber et al., 2024; Basant et al., 2025) has generally favored configurations with a high proportion of Mamba layers—for example, a 1:7 ratio. We revisit this design choice and offer deeper insights by comparing various ratios from both quality and efficiency perspectives.

**Positioning of computational primitives.** Another open question is whether the position of interleaved blocks affects model quality, and how best to arrange them for optimal performance. There is currently no clear consensus on optimal positioning, as it may depend on factors such as block ratio and model scale. Although prior work (Lieber et al., 2024; Ren et al., 2025; Basant et al., 2025) often places Transformer blocks in the middle layers, comprehensive studies on this topic are still lacking. To fill this gap, we conduct extensive ablation experiments, varying the positions of Transformer blocks (e.g., early, middle, and late layers) across different ratios and model sizes.

## 3.2 INTRA-LAYER HYBRID MODEL

**Definition.** Intra-layer hybrid models achieve fine-grained fusion within individual layers by blending softmax attention and linear attention in parallel. A common approach (Dong et al., 2024; Zuo et al., 2025; Xiao et al., 2025) is head-wise splitting, where some heads use Transformer attention while others use Mamba. Here, we use intra-hybrid and Mamba blocks as primitive candidates, so that this includes an interleaving mechanism akin to inter-layer hybrid models.

**Related works.** Intra-layer hybrid models can be classified by how each token interacts with different modules. In the head-wise splitting approach (e.g., Hymba (Dong et al., 2024), Falcon-H1 (Zuo et al., 2025), WuNeng (Xiao et al., 2025), and Dragon), attention heads are divided into groups, with each group assigned to a distinct module. Alternatively, works like Liger (Lan et al., 2025) process each token through all modules in parallel and fuse the outputs. For sequence-wise splitting, different modules are applied to context tokens based on their positions when computing attention scores. This splitting can be determined by an absolute point in the sequence, as in TransMamba (Li et al., 2025b), or by relative position, as in LoLCATs (Zhang et al., 2024). While not hybrid, differential architectures such as Differential Transformer (Ye et al., 2024) and Differential Mamba (Schneider et al., 2025) also use head-wise splitting, but use the same primitive type and subtract their attention scores.

Table 2: **Hybrid models achieve better quality than baselines under equal data and compute constraints.** $N_{\text{hyb}}$ denotes the number of intra-hybrid block primitive. We calculate total training compute FLOPs and cache sizes for a sequence length of 8K tokens. We use an approximate block ratio of 1:5 for SWA and hybrid architectures, which mix two different types of block.

| | Base Config | | | | | Cost | | | NLL (↓) | | Few-shot Accuracy (↑) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Models** | N-emb | $N_{\text{attn}}$ | $N_{\text{swa}}$ | $N_{\text{ssm}}$ | $N_{\text{hyb}}$ | Tok | FLOPs | Cache | DCLM | PG19 | LD | HS | PQ | ARC | OB | Avg |
| Llama | 0.97B | 16 | - | - | - | 60B | 4.5 e20 | 256 | 2.750 | 2.875 | 56.1 | 55.2 | 72.8 | 43.2 | 32.7 | 52.0 |
| SWA | 0.97B | 3 | 13 | - | - | 60B | 3.8 e20 | 63 | 2.741 | 2.867 | 56.0 | 55.9 | 72.6 | 44.2 | 34.8 | 52.7 |
| Mamba | 0.99B | - | - | 13 | - | 60B | 3.7 e20 | 13 | 2.758 | 2.891 | 53.7 | 55.1 | **73.8** | 43.9 | 35.2 | 52.3 |
| Inter-H | 0.96B | 2 | - | 11 | - | 60B | 3.7 e20 | 43 | 2.735 | 2.861 | 56.4 | 55.4 | 73.1 | 44.8 | **36.6** | 53.3 |
| Intra-H | 0.98B | - | - | 11 | 2 | 60B | 3.7 e20 | 38 | **2.728** | **2.853** | **58.7** | **57.4** | 73.3 | **47.9** | 36.4 | **54.7** |
| SWA | 0.97B | 3 | 13 | - | - | 71B | 4.5 e20 | 63 | 2.724 | 2.845 | 57.0 | 56.9 | 73.1 | 46.0 | 36.0 | 53.8 |
| Mamba | 0.97B | - | - | 13 | - | 73B | 4.5 e20 | 13 | 2.736 | 2.865 | 55.2 | 57.1 | 73.8 | 45.3 | 36.0 | 53.5 |
| Inter-H | 0.96B | 2 | - | 11 | - | 73B | 4.5 e20 | 43 | 2.716 | 2.842 | 56.7 | 57.6 | 73.3 | 46.5 | 35.8 | 54.0 |
| Intra-H | 0.98B | - | - | 11 | 2 | 72B | 4.5 e20 | 38 | **2.709** | **2.831** | **58.4** | **57.7** | **74.4** | **46.9** | **37.0** | **54.9** |

**Architectural variants for intra-layer hybrid.** We adapt a head-wise splitting approach, assigning different primitives to two groups of attention heads. Especially, query and key states in Transformer are projected to reduced dimensions, while the value state is expanded back to the original size (Ye et al., 2024; Schneider et al., 2025). For Mamba, the hidden dimension of SSMs is similarly reduced based on configuration. Various fusion designs are explored, including normalization strategies—with or without group normalization (Wu & He, 2018) as in Hymba (Dong et al., 2024)); learnable scalar—such as scaling (Dong et al., 2024; Ye et al., 2024; Schneider et al., 2025) or gating parameters (Lan et al., 2025; Xiao et al., 2025); fusion operations—including addition (Dong et al., 2024; Lan et al., 2025; Xiao et al., 2025), subtraction (Ye et al., 2024; Schneider et al., 2025), or concatenation (Xiao et al., 2025); and the number of output projection. We thoroughly evaluate these variants at both 350M and 1B scales, comparing their performance to existing intra-hybrid architectures.

**Dimension ratios in intra-hybrid block.** The ratio of parameter sizes between Transformer and Mamba modules within intra-hybrid blocks—controlled by their hidden dimension allocations—is one of key architectural considerations. By observing how model quality changes as this ratio varies, we can infer the relative importance of each primitive. Furthermore, assuming expert parallelism (Rajbhandari et al., 2022) enables parallel execution of the modules, overall efficiency will be bounded by the slower Transformer component. Thus, we investigate how much we can reduce the dimension assigned to the Transformer, aiming to enhance efficiency without compromising overall quality.

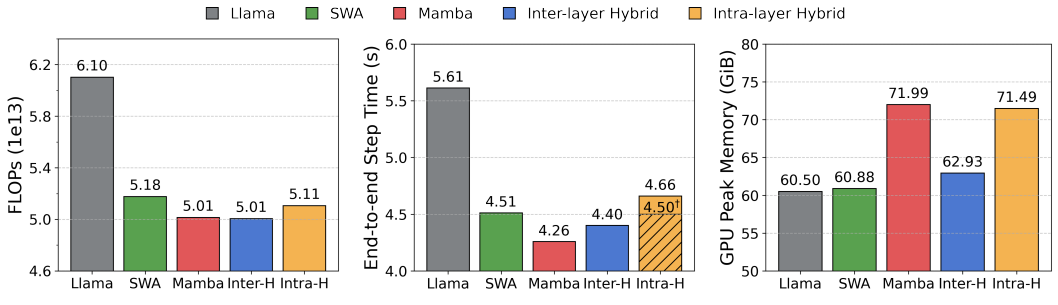
**Block ratios and positioning of primitives.** In our intra-layer hybrid, each block is either an intra-hybrid block or a Mamba block. By varying the block ratio to control the interpolation degree, we analyze how different configurations affect both quality and efficiency, aiming to understand the trade-offs in intra-hybridization. We further examine how the placement of these intra-hybrid blocks at different depths influences model quality. Since both Transformer and Mamba are included, unlike using homogeneous primitives, we explore how its behavior differs from that of inter-layer hybrid models.

## 4 Experiments

We build hybrid models with computational primitives following the configurations of Llama 3.2 (Llama Team, 2024) and Mamba 2 (Dao & Gu, 2024) architectures. We use the TorchTitan framework (Liang et al., 2024) for large-scale LLM training with H200 GPUs. See Appendix F for further details.

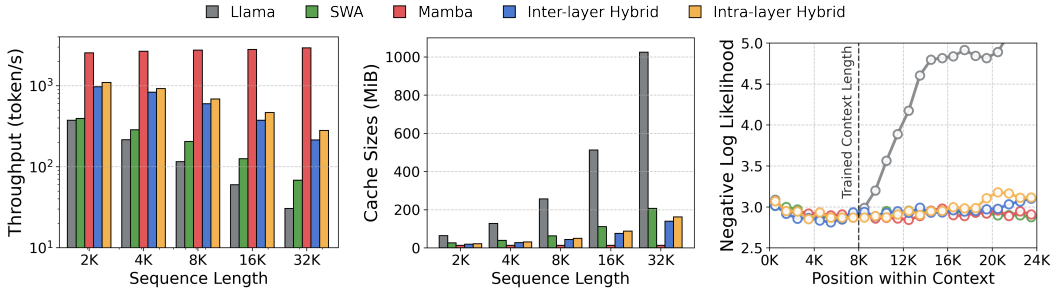
### 4.1 Main Results on Quality

**Hybrid architectures significantly outperform homogeneous models.** Table 2 compares our two hybridization strategies—optimized along positioning and design choice axes (see §4.5 and §4.6)—with conventional baselines (Llama Team, 2024; Dao & Gu, 2024; Gemma Team et al., 2025). Our controlled experiments show that hybrid models, including SWA, consistently outperform homogeneous models in negative log-likelihood (NLL) and few-shot accuracy under the same data budget. This demonstrates that effectively combining complementary inductive biases from different primitives is key to improving model quality through hybridization. Notably, we provide




(a) FLOPs per sample (↓) (b) End-to-end step time (↓) (c) Memory need (↓)

Figure 2: **Hybrid models have lower FLOPs, which directly leads to reduced actual training time.** We measure metrics for 1B model using 8 H200 GPUs with FSDP, `torch.compile`, 8K lengths, and a local batch size of 4, without activation checkpointing. Both SWA and hybrid models use a 1:5 block ratio. † indicates a theoretical time achievable with parallelism (Rajbhandari et al., 2022).




(a) Throughput (↑) (b) Cache sizes (↓) (c) NLL on PG19 (↓)

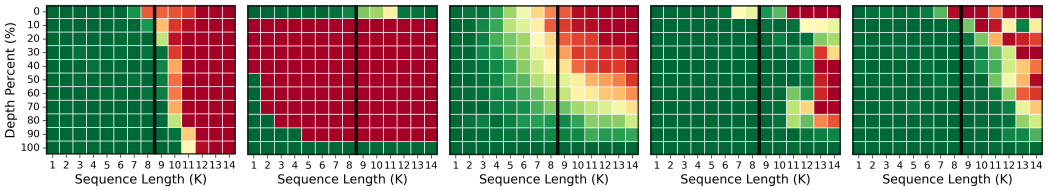
Figure 3: **(a, b) Hybrid architectures show sub-quadratic scaling of inference throughput and memory as sequence increases.** SWA and hybrid 1B models use block ratio of 1:5. For throughput, we set the prompt length to 512 and the batch size to 4. **(c) Mamba enables length generalization in terms of perplexity, allowing hybrids to maintain strong performance.** We use 1B models trained with a compute budget of 4.5e20 and 8K lengths. Loss is averaged every 1K positions over 30 samples.

a meaningful comparison between inter- and intra-hybrid architectures under matched budgets, which has not been previously explored or shared (Ren et al., 2024; Jamba Team et al., 2024; Dong et al., 2024; Basant et al., 2025). Under the same FLOP budget, hybrids achieve even more substantial quality gains (e.g., 2.9% increase in accuracy and 0.04 reduction in NLL). We find that these advantages are consistent across different block ratios and model scales, demonstrating the robustness of the hybrid approach (see Figure 1b).

### 4.2 Main Results on Efficiency

**FLOPs reduction in hybrid models translate into faster actual end-to-end training time.** Mamba's linear complexity leads to lower FLOPs than Transformer (18% lower for 1B scale at 8K lengths; see Table 1 for details). As shown in Figure 2, hybrid models leverage these advantage for superior performance in compute-bound scenarios. Empirically, lower FLOPs almost directly yield faster training, aided by parallel scan algorithms (Blelloch, 1990; Smith et al., 2022). For intra-hybrid models, the expert parallelism (Rajbhandari et al., 2022) can theoretically optimize training speed further. Although Mamba and hybrid models may use slightly more memory during training, adjusting the block ratio provides flexibility to balance between efficiency and model quality.

**Hybrid models achieve a superior Pareto frontier of inference throughput and quality.** Figures 3a and 3b demonstrate that hybrid architectures, by leveraging Mamba's linear complexity and constant cache size, sustain high throughput and sub-quadratic memory growth across context lengths. This enables faster generation compared to baselines, while still maintaining high output quality (see Figure 1b). Although SWA uses a 512 + 64 (window + sink) attention map, it remains slower than Mamba's linear attention, making hybrids overall faster. Interestingly, intra-layer hybrids, which utilize half-sized Transformer, further outperform inter-layer hybrids even when executed sequentially.

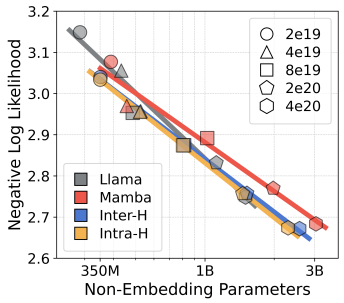



(a) Llama (b) Sliding window (c) Mamba (d) Inter-hybrid (e) Intra-hybrid

Figure 4: **Hybrid models overcome the limitations of both foundational primitives, achieving superior in-context retrieval performance.** We insert a needle (random 7-digit number associated with random city name (Gemini Team et al., 2024)) across the 0–100% depth range (y-axis) for context lengths up to 14K (x-axis). Over 100 trials, green indicates 100% accuracy, while red denotes 0% accuracy. We use 1B model checkpoints trained with 8K length and a FLOPs budget of 4.5e20. SWA and hybrid models use 1:5 block ratio.

Table 3: **(*Left*) All architectures consistently achieve significant quality gains from MoE integration.** All models are trained on 60B tokens, with hybrid models using a 1:5 block ratio. We use one shared expert and the selected top-1 expert among eight. Additionally, a token-choice router with loss-free balancing algorithm is utilized (Wang et al., 2024b; DeepSeek-AI, 2024). **(*Right*) Hybrid architectures demonstrates a compute-optimal scaling line that lies between those of Transformer and Mamba.** We train models at four different scales—100M, 350M, 1B, and 3B—across five compute budgets. Each marker indicates the optimal model size for a given compute budget. For hybrid models, we use a 1:5 block ratio.

| | Base Config | | | | | | MoE | | Performance (↓ / ↓ / ↑) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Models** | Act | Total | $N_{\text{attn}}$ | $N_{\text{ssm}}$ | $N_{\text{hyb}}$ | FLOPs | Num | Act | DCLM | PG19 | Acc |
| Llama | 0.97B | 0.97B | 16 | - | - | 4.5 e20 | - | - | 2.750 | 2.875 | 52.0 |
| | 0.97B | 3.79B | 16 | - | - | 4.5 e20 | 1+8 | 1+1 | **2.656** | **2.775** | **55.8** |
| Mamba | 0.99B | 0.99B | - | 13 | - | 3.7 e20 | - | - | 2.758 | 2.891 | 52.3 |
| | 0.99B | 3.28B | - | 13 | - | 3.7 e20 | 1+8 | 1+1 | **2.673** | **2.803** | **55.0** |
| Inter-H | 0.96B | 0.96B | 2 | 11 | - | 3.7 e20 | - | - | 2.735 | 2.861 | 53.3 |
| | 0.96B | 3.25B | 2 | 11 | - | 3.7 e20 | 1+8 | 1+1 | **2.653** | **2.780** | **56.0** |
| Intra-H | 0.98B | 0.98B | - | 11 | 2 | 3.7 e20 | - | - | 2.728 | 2.853 | 54.7 |
| | 0.98B | 3.27B | - | 11 | 2 | 3.7 e20 | 1+8 | 1+1 | **2.648** | **2.775** | **56.9** |




### 4.3 Long-context Capability Evaluation

**Position-wise loss on PG19 shows hybrid-Mamba models extrapolates well to longer contexts.** Figure 3c compares position-wise loss on the PG19 corpus (Rae et al., 2019a) to examine the long-context capabilities of various architectures. All models show decreasing NLL up to the 8K-token pretraining context, as later tokens benefit from more context. Beyond 8K, SSMs can effectively extrapolate (Gu & Dao, 2023; Dao & Gu, 2024; Yang et al., 2024), while Transformer struggles due to its positional encoding methods (Press et al., 2021; Kazemnejad et al., 2023; Zhou et al., 2024). Therefore, both hybrid architectures with a 1:5 block ratio benefit more from Mamba, demonstrating stronger length generalization. This characteristic will vary depending on the interpolation degree determined by the block ratio.

**Hybrid models overcome weakness of Transformer and Mamba in in-context retrieval.** Figure 4 presents in-context retrieval performance on the Needle-In-A-Haystack benchmark (Kamradt, 2023; Kuratov et al., 2024). A needle is inserted at a specific depth in the context, and models are evaluated on their ability to retrieve (generate) it. Transformer accuracy drops to near zero beyond 8K context length, as expected. On the other hand, SWA and Mamba also struggle with retrieval outside their local window and sink token regions, or in long-range tokens, despite strong perplexity scores in Figure 3c; this is likely due to their focus on local information (Ben-Kish et al., 2024). In contrast, both inter- and intra-hybrid models surprisingly maintain strong retrieval performance up to about 1.5x the pretraining length, overcoming the limitations of the base primitives rather than simply inheriting them. While retrieval accuracy in the middle of extrapolated contexts does decline (Liu et al., 2023), hybrid models consistently demonstrate improved retrieval capabilities.

Table 4: (***Left***) **Inter-layer hybrid achieves the best quality at a 1:1 block ratio, but the 1:5 ratio is preferable to balance efficiency and quality.** Transformer blocks are evenly distributed in the middle. All models are trained on 60B tokens. (***Right***) **Interleaving Transformer blocks at intermediate depths is key for optimal performance.** The upper figure shows results of ablating the position of a single Transformer block in 1B (13 layers) and 350M (11 layers) models with a 1:12 ratio. In the lower figure, after distributing Transformer blocks in the middle of 1B model, we move the first block to the first layer (Front) or the last block to the last layer (End). All models are trained on 60B tokens.

| Sizes | Ratio | Base Config | | | Performance (↓ / ↓ / ↑) | | |
|---|---|---|---|---|---|---|---|
| | | N-emb | $N_{attn}$ | $N_{ssm}$ | DCLM | PG19 | Acc |
| 1B | 1 : 0 | 0.97B | 16 | - | 2.750 | 2.875 | 52.0 |
| | 0 : 1 | 0.99B | - | 13 | 2.758 | 2.891 | 52.3 |
| | 1 : 1 | 0.96B | 7 | 7 | **2.725** | **2.847** | **54.0** |
| | 1 : 3 | 0.94B | 3 | 10 | 2.732 | 2.857 | 53.8 |
| | 1 : 5 | 0.96B | 2 | 11 | 2.735 | 2.861 | 53.3 |
| | 1 : 12 | 0.97B | 1 | 12 | 2.741 | 2.866 | 53.1 |
| 350M | 1 : 0 | 0.35B | 14 | - | 2.882 | 3.015 | 48.7 |
| | 0 : 1 | 0.35B | - | 11 | 2.880 | 3.024 | 48.7 |
| | 1 : 1 | 0.35B | 6 | 6 | **2.850** | **2.985** | 49.3 |
| | 1 : 3 | 0.34B | 3 | 8 | 2.858 | 2.994 | **50.2** |
| | 1 : 5 | 0.35B | 2 | 9 | 2.860 | 2.999 | 49.4 |
| | 1 : 12 | 0.36B | 1 | 10 | 2.864 | 3.003 | 49.3 |


### 4.4 Scaling Analysis

**Mixture-of-Experts are fully compatible with hybrid architectures.** A few recent inter-layer hybrid models (Lieber et al., 2024; Jamba Team et al., 2024) have incorporated Mixture-of-Experts (MoE) (Shazeer et al., 2017; Fedus et al., 2022) to improve performance at fixed compute. In Table 3, we re-examine MoE in inter-layer hybrids and also evaluate intra-layer hybridization, comparing both to baseline models. Across all architectures, MoE yields a substantial reduction in NLL (by 0.08) and 4 point increase in few-shot accuracy. Since hybridization is applied to the attention component, integrating MoE into the FFN layer remains compatible with all hybrid models. The data-hungry nature of MoE also enables more efficient scaling of hybrid models for a fixed number of activated parameters.

**Hybrid models are efficient and scalable architectures.** The figure to the right of Table 3 analyzes the scalability of different model architectures and identifies compute-optimal scaling strategies (Kaplan et al., 2020; Hoffmann et al., 2022). The compute-optimal line illustrates the best achievable quality and parameter sizes for each architecture under a given computational budget. Mamba performs best with larger models and less data feeding, while Transformers favor a higher token-to-parameter ratio of around 20 (Hoffmann et al., 2022). Hybrid models show intermediate scaling behavior, with intra-layer hybrids being a little slightly more data-hungry. These results provide clear guidance on how to optimally scale up hybrid architectures.

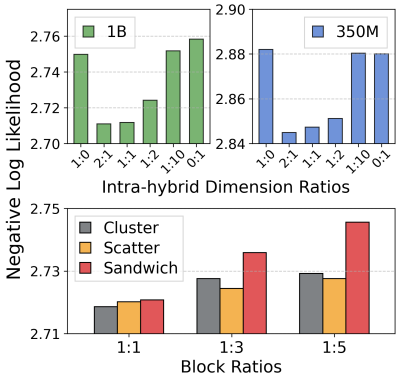
### 4.5 Ablation Studies for Inter-layer Hybridization

**To ensure quality, aim for a high 1:1 ratio, but to balance with efficiency, use about 1:5 ratio.** In Table 4, we compare inter-hybrid models with six block ratios at two scales (1:0 = Transformer, 0:1 = Mamba). Hybrid architectures consistently outperform homogeneous ones, with the balanced 1:1 ratio yielding the best quality. However, due to the Transformer's quadratic complexity, inference throughput gains become limited. To balance both efficiency and quality, a ratio of around 1:5 appears to be optimal, which aligns with the lower ratios (e.g., 1:5, 1:7) adopted by large hybrid language models (Lieber et al., 2024; Wang et al., 2025; Basant et al., 2025).

**Never place Transformer blocks at the front.** Deciding the order of Transformer and Mamba blocks is a key design choice. Our ablation study (see figures next to Table 4) shows that, for a 1:12 ratio, placing the Transformer block in the early layers leads to significant performance drop, while positioning it in the middle yields the best results. Similar experiments with higher block ratios (bottom-right figure) also confirm that putting Transformer blocks at the front consistently leads to worse performance than homogeneous models, regardless of ratio. These finding support prior work favoring middle placement of Transformer (Park et al., 2024; Jamba Team et al., 2024; Dong et al., 2024). In the future, more efficient methods like layer-wise sensitivity analysis (Yang et al., 2025) or architecture search (Thomas et al., 2024) may help optimize block positions.

Table 5: **(*Left*) We identify a more optimal architecture than previous designs through ablation studies on intra-hybrid block.** We train a 1B model on 60B tokens, replacing all layers with intra-hybrid blocks (i.e., 1:0 block ratio). All heads are split into two, with each linked to a half-sized primitive. † denotes our re-implementation of the two prior works, equipped with value dimension expansion for GQA. **(*Right*) Keeping larger Transformer dimension within intra-hybrid block improves quality (despite reduced efficiency), and placing the intra-hybrid block in the middle yields the best results.** All 1B models with the optimal architecture from left, are trained on 60B tokens. In the figure above, we use a block ratio of 1:0, where all layers become intra-hybrid blocks. For lower figure, we place intra-hybrid blocks in the middle of 1B model by: placing consecutively (Cluster), distributing evenly (Scatter), or placing at the beginning and end as well (Sandwich).

| | Design Choices | | | | | Performance (↓ / ↓ / ↑) | | |
|---|---|---|---|---|---|---|---|---|
| **Models** | Module | Norm | Scalar | Fusion | Out | DCLM | PG19 | Acc |
| Llama | T | - | - | - | - | 2.750 | 2.875 | 52.0 |
| Mamba | M | - | - | - | - | 2.758 | 2.891 | 52.3 |
| Diff-T | T / T | - | Diff-T | Diff | Joint | 2.727 | 2.848 | 53.6 |
| Diff-M | M / M | - | Diff-M | Diff | Sep | 2.759 | 2.892 | 53.0 |
| Hymba† | T / M | Group | Scale | Add | Joint | 2.726 | 2.846 | 52.4 |
| Falcon-H1† | T / M | - | - | Conc | Joint | 2.720 | 2.840 | 53.6 |
| Variants | T / M | - | Scale | Add | Joint | 2.740 | 2.865 | 53.9 |
| | T / M | Group | - | Add | Joint | 2.721 | 2.840 | 54.5 |
| | T / M | Group | Gate | Add | Joint | 2.751 | 2.876 | 53.1 |
| | T / M | Group | - | Diff | Joint | 2.721 | 2.842 | 54.0 |
| | T / M | Group | - | Conc | Joint | 2.715 | 2.833 | 54.8 |
| | T / M | Group | - | Add | Sep | 2.714 | 2.834 | 54.5 |
| | T / M | Group | - | Diff | Sep | **2.712** | **2.831** | **54.9** |




### 4.6 Ablation Studies for Intra-layer Hybridization

**Novel outperforming architectural variant for intra-hybrid block.** In designing intra-hybrid blocks, we explore four axes: normalization layer, learnable scalar, fusion operation, and output projection. In Table 5, our experiments reveal that normalization is crucial due to the scale differences between modules (Dong et al., 2024), which makes additional scaling factors unnecessary. For output fusion, either subtracting the outputs to mitigate attention noise or simply concatenating them yield the best quality. The resulting architecture surpasses previous intra-hybrid models (e.g., Hymba (Dong et al., 2024), Falcon-H1 (Zuo et al., 2025)), and hybridizing Transformer and Mamba proves superior in both quality and efficiency over differential architectures that use same type of primitives (e.g., Differential Transformer (Ye et al., 2024), Differential Mamba (Schneider et al., 2025)).

**Enlarging Transformer dimension improves quality, despite reduced efficiency.** The upper-right figure presents ablation results for dimension allocation ratios within the intra-hybrid block, defined by the proportion of query / key dimension in Transformer versus pre-expansion dimension in Mamba (with 1:0 and 0:1 representing pure Transformer and Mamba, respectively). The results indicate that a balanced allocation is important for quality, with larger Transformer dimensions leading to greater performance gains (similar observations in Zuo et al. (2025))—suggesting that the Transformer component even plays a more critical role than Mamba. However, since throughput is limited by the Transformer under parallel execution, 1:1 dimension ratio offers a practical and effective balance.

**Evenly scattering intra-hybrid blocks across depths yields the best quality.** We further investigate block ratio and ordering strategies for intra-hybrid models (see lower figure next to Table 5). Increasing the proportion of Transformer-containing intra-hybrid blocks consistently improves quality, in line with previous findings on primitive importance from dimension ratio ablations. However, using more Mamba blocks remains a practical choice for efficiency. For block positioning, motivated by lessons from §4.5, we keep intra-hybrid blocks in the middle positions for ablation studies. Notably, evenly distributing these intra-hybrid blocks across depths yields the best results, while placing them at the ends (Sandwich strategy) leads to huge performance drops.

## 5 Conclusion

In this work, we present a holistic analysis for inter-layer and intra-layer hybrid architectures. Our comprehensive evaluation demonstrates that hybrid models consistently outperform homogeneous architectures across multiple quality metrics. Notably, they also achieve superior efficiency, achieving faster training and inference, much more than widely adopted sliding window attention models. Our findings shed practical guidance for designing high-quality, efficient hybrid architectures.

## REFERENCES


Bilge Acun, Prasoon Sinha, Newsha Ardalani, Sangmin Bae, Alicia Golden, Chien-Yu Lin, Meghana Madhyastha, Fei Sun, Neeraja J Yadwadkar, and Carole-Jean Wu. Composer: A search framework for hybrid neural architecture design. *arXiv preprint arXiv:2510.00379*, 2025.

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

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pp. 929–947, 2024.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Aarti Basant, Abhijit Khairnar, Abhijit Paithankar, Abhinav Khattar, Adi Renduchintala, Adithya Renduchintala, Aditya Malte, Akhiad Bercovich, Akshay Hazare, Alejandra Rico, et al. Nvidia nemotron nano 2: An accurate and efficient hybrid mamba-transformer reasoning model. *arXiv preprint arXiv:2508.14444*, 2025.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*, 2024.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Assaf Ben-Kish, Itamar Zimerman, Shady Abu-Hussein, Nadav Cohen, Amir Globerson, Lior Wolf, and Raja Giryes. Decimamba: Exploring the length extrapolation potential of mamba. *arXiv preprint arXiv:2406.14528*, 2024.

Aviv Bick, Kevin Li, Eric Xing, J Zico Kolter, and Albert Gu. Transformers to ssms: Distilling quadratic knowledge to subquadratic models. *Advances in Neural Information Processing Systems*, 37:31788–31812, 2024.

Guy E Blelloch. Prefix sums and their applications. 1990.

Aleksandar Botev, Soham De, Samuel L Smith, Anushan Fernando, George-Cristian Muraru, Ruba Haroun, Leonard Berrada, Razvan Pascanu, Pier Giuseppe Sessa, Robert Dadashi, et al. Recurrentgemma: Moving past transformers for efficient open language models. *arXiv preprint arXiv:2404.07839*, 2024.

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

Team Cohere, Arash Ahmadian, Marwan Ahmed, Jay Alammar, Milad Alizadeh, Yazeed Alnumay, Sophia Althammer, Arkady Arkhangorodsky, Viraat Aryabumi, Dennis Aumiller, et al. Command a: An enterprise-ready large language model. *arXiv preprint arXiv:2504.00698*, 2025.

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

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

Soham De, Samuel L Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, et al. Griffin: Mixing gated linear recurrences with local attention for efficient language models. *arXiv preprint arXiv:2402.19427*, 2024.

DeepSeek-AI. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 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, et al. Hymba: A hybrid-head architecture for small language models. *arXiv preprint arXiv:2411.13676*, 2024.

Alexander R. Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R. Radev. Multi-news: a large-scale multi-document summarization dataset and abstractive hierarchical model, 2019.

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

Daniel Y Fu, Tri Dao, Khaled K Saab, Armin W Thomas, Atri Rudra, and Christopher Ré. Hungry hungry hippos: Towards language modeling with state space models. *arXiv preprint arXiv:2212.14052*, 2022.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Google Gemini Team. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

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

Google Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Paolo Glorioso, Quentin Anthony, Yury Tokpanov, Anna Golubeva, Vasudev Shyam, James Whittington, Jonathan Pilault, and Beren Millidge. The zamba2 suite: Technical report. *arXiv preprint arXiv:2411.15242*, 2024a.

Paolo Glorioso, Quentin Anthony, Yury Tokpanov, James Whittington, Jonathan Pilault, Adam Ibrahim, and Beren Millidge. Zamba: A compact 7b ssm hybrid model. *arXiv preprint arXiv:2405.16712*, 2024b.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33: 1474–1487, 2020.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021a.

Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré. Combining recurrent, convolutional, and continuous-time models with linear state space layers. *Advances in neural information processing systems*, 34:572–585, 2021b.

Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. *Advances in Neural Information Processing Systems*, 35:35971–35983, 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. *arXiv preprint arXiv:2508.15884*, 2025.

Namgyu Ho, Sangmin Bae, Taehyeon Kim, Hyunjik Jo, Yireun Kim, Tal Schuster, Adam Fisch, James Thorne, and Se-Young Yun. Block transformer: Global-to-local language modeling for fast inference. *Advances in Neural Information Processing Systems*, 37:48740–48783, 2024.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Haowen Hou, Zhiyi Huang, Kaifeng Tan, Rongchang Lu, and Fei Richard Yu. Rwkv-x: A linear complexity hybrid language model. *arXiv preprint arXiv:2504.21463*, 2025.

Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. Transformer quality in linear time. In *International conference on machine learning*, pp. 9099–9117. PMLR, 2022.

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

Tencent Hunyuan Team, Ao Liu, Botong Zhou, Can Xu, Chayse Zhou, ChenChen Zhang, Chengcheng Xu, Chenhao Wang, Decheng Wu, Dengpeng Wu, et al. Hunyuan-turbos: Advancing large language models through mamba-transformer synergy and adaptive chain-of-thought. *arXiv preprint arXiv:2505.15431*, 2025.

Ai2 Jamba Team, Barak Lenz, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, et al. Jamba-1.5: Hybrid transformer-mamba models at scale. *arXiv preprint arXiv:2408.12570*, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. triviaqa: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. *arXiv e-prints*, art. arXiv:1705.03551, 2017.

Gregory Kamradt. NeedleInAHaystack: A repository for testing LLMs. https://github.com/gkamradt/LLMTest_NeedleInAHaystack/blob/main/README.md, 2023. Accessed: 2023-10-31.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36:24892–24928, 2023.

MoonshotAI Kimi Team. Kimi linear: An expressive, efficient attention architecture, 2025. URL https://arxiv.org/abs/2510.26692.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018. doi: 10.1162/tacl_a_00023. URL https://aclanthology.org/Q18-1023.

Yury Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *Advances in Neural Information Processing Systems*, 37:106519–106554, 2024.

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.

Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025a.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Guha, Sedrick Scott Keh, Kushal Arora, et al. Datacomp-lm: In search of the next generation of training sets for language models. *Advances in Neural Information Processing Systems*, 37:14200–14282, 2024.

Yixing Li, Ruobing Xie, Zhen Yang, Xingwu Sun, Shuaipeng Li, Weidong Han, Zhanhui Kang, Yu Cheng, Chengzhong Xu, Di Wang, et al. Transmamba: Flexibly switching between transformer and mamba. *arXiv preprint arXiv:2503.24067*, 2025b.

Wanchao Liang, Tianyu Liu, Less Wright, Will Constable, Andrew Gu, Chien-Chin Huang, Iris Zhang, Wei Feng, Howard Huang, Junjie Wang, et al. Torchtitan: One-stop pytorch native solution for production ready llm pre-training. *arXiv preprint arXiv:2410.06511*, 2024.

Opher Lieber, Barak Lenz, Hofit Bata, Gal Cohen, Jhonathan Osin, Itay Dalmedigos, Erez Safahi, Shaked Meirom, Yonatan Belinkov, Shai Shalev-Shwartz, et al. Jamba: A hybrid transformer-mamba language model. *arXiv preprint arXiv:2403.19887*, 2024.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*, 2023.

Meta Llama Team. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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

Enzhe Lu, Zhejun Jiang, Jingyuan Liu, Yulun Du, Tao Jiang, Chao Hong, Shaowei Liu, Weiran He, Enming Yuan, Yuzhi Wang, et al. Moba: Mixture of block attention for long-context llms. *arXiv preprint arXiv:2502.13189*, 2025.

Microsoft Research. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

OpenaAI, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

OpenAI, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Artidoro Pagnoni, Ram Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason Weston, Luke Zettlemoyer, et al. Byte latent transformer: Patches scale better than tokens. *arXiv preprint arXiv:2412.09871*, 2024.

Jongho Park, Jaeseung Park, Zheyang Xiong, Nayoung Lee, Jaewoong Cho, Samet Oymak, Kangwook Lee, and Dimitris Papailiopoulos. Can mamba learn how to learn? a comparative study on in-context learning tasks. *arXiv preprint arXiv:2402.04248*, 2024.

Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Xingjian Du, Haowen Hou, Jiaju Lin, Jiaxing Liu, Janna Lu, William Merrill, et al. Rwkv-7" goose" with expressive dynamic state evolution. *arXiv preprint arXiv:2503.14456*, 2025.

Michael Poli, Armin W Thomas, Eric Nguyen, Pragaash Ponnusamy, Björn Deiseroth, Kristian Kersting, Taiji Suzuki, Brian Hie, Stefano Ermon, Christopher Ré, et al. Mechanistic design and scaling of hybrid architectures. *arXiv preprint arXiv:2403.17844*, 2024.

Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

Zhen Qin, Xiaodong Han, Weixuan Sun, Dongxu Li, Lingpeng Kong, Nick Barnes, and Yiran Zhong. The devil in linear transformer. *arXiv preprint arXiv:2210.10340*, 2022.

Zhen Qin, Weigao Sun, Dong Li, Xuyang Shen, Weixuan Sun, and Yiran Zhong. Lightning attention-2: A free lunch for handling unlimited sequence lengths in large language models. *arXiv preprint arXiv:2401.04658*, 2024.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, et al. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. *arXiv preprint arXiv:2505.06708*, 2025.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint*, 2019a. URL https://arxiv.org/abs/1911.05507.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019b.

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264.

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. *arXiv preprint arXiv:2406.07522*, 2024.

Liliang Ren, Congcong Chen, Haoran Xu, Young Jin Kim, Adam Atkinson, Zheng Zhan, Jiankai Sun, Baolin Peng, Liyuan Liu, Shuohang Wang, et al. Decoder-hybrid-decoder architecture for efficient reasoning with long generation. *arXiv preprint arXiv:2507.06607*, 2025.

Nadav Schneider, Itamar Zimerman, and Eliya Nachmani. Differential mamba. *arXiv preprint arXiv:2507.06204*, 2025.

Eva Sharma, Chen Li, and Lu Wang. BIGPATENT: A large-scale dataset for abstractive and coherent summarization. *CoRR*, abs/1906.03741, 2019. URL http://arxiv.org/abs/1906.03741.

Noam Shazeer. GLU variants improve transformer. *CoRR*, abs/2002.05202, 2020. URL https://arxiv.org/abs/2002.05202.

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

Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. Simplified state space layers for sequence modeling. *arXiv preprint arXiv:2208.04933*, 2022.

Weigao Sun, Jiaxi Hu, Yucheng Zhou, Jusen Du, Disen Lan, Kexin Wang, Tong Zhu, Xiaoye Qu, Yu Zhang, Xiaoyu Mo, et al. Speed always wins: A survey on efficient architectures for large language models. *arXiv preprint arXiv:2508.09834*, 2025.

Armin W Thomas, Rom Parnichkun, Alexander Amini, Stefano Massaroli, and Michael Poli. Star: Synthesis of tailored architectures. *arXiv preprint arXiv:2411.17800*, 2024.

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

Roger Waleffe, Wonmin Byeon, Duncan Riach, Brandon Norick, Vijay Korthikanti, Tri Dao, Albert Gu, Ali Hatamizadeh, Sudhakar Singh, Deepak Narayanan, et al. An empirical study of mamba-based language models. *arXiv preprint arXiv:2406.07887*, 2024.

Dustin Wang, Rui-Jie Zhu, Steven Abreu, Yong Shan, Taylor Kergan, Yuqi Pan, Yuhong Chou, Zheng Li, Ge Zhang, Wenhao Huang, et al. A systematic analysis of hybrid linear attention. *arXiv preprint arXiv:2507.06457*, 2025.

Junxiong Wang, Daniele Paliotta, Avner May, Alexander Rush, and Tri Dao. The mamba in the llama: Distilling and accelerating hybrid models. *Advances in Neural Information Processing Systems*, 37:62432–62457, 2024a.

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

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

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

Liu Xiao, Li Zhiyuan, and Lin Yueyu. Wuneng: Hybrid state with attention. *arXiv preprint arXiv:2504.19191*, 2025.

Chen Xing, Devansh Arpit, Christos Tsirigotis, and Yoshua Bengio. A walk with sgd. *arXiv preprint arXiv:1802.08770*, 2018.

Mingyu Yang, Mehdi Rezagholizadeh, Guihong Li, Vikram Appia, and Emad Barsoum. Zebra-llama: Towards extremely efficient hybrid models. *arXiv preprint arXiv:2505.17272*, 2025.

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

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential transformer. *arXiv preprint arXiv:2410.05258*, 2024.

Lili Yu, Dániel Simig, Colin Flaherty, Armen Aghajanyan, Luke Zettlemoyer, and Mike Lewis. Megabyte: Predicting million-byte sequences with multiscale transformers. *Advances in Neural Information Processing Systems*, 36:78808–78823, 2023.

Michael Zhang, Simran Arora, Rahul Chalamala, Alan Wu, Benjamin Spector, Aaryan Singhal, Krithik Ramesh, and Christopher Ré. Lolcats: On low-rank linearizing of large language models. *arXiv preprint arXiv:2410.10254*, 2024.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

Yongchao Zhou, Uri Alon, Xinyun Chen, Xuezhi Wang, Rishabh Agarwal, and Denny Zhou. Transformers can achieve length generalization but not robustly. *arXiv preprint arXiv:2402.09371*, 2024.

Jingwei Zuo, Maksim Velikanov, Ilyas Chahed, Younes Belkada, Dhia Eddine Rhayem, Guillaume Kunsch, Hakim Hacid, Hamza Yous, Brahim Farhat, Ibrahim Khadraoui, et al. Falcon-h1: A family of hybrid-head language models redefining efficiency and performance. *arXiv preprint arXiv:2507.22448*, 2025.

## Contents

## A The Use of Large Language Models

We used large language models based on Llama 3.2 (Llama Team, 2024) and Llama 4 to polish the overall writing after drafting the paper ourselves. Additionally, we utilized the same models for vibe coding when fixing bugs and when making the initial drafts of the figures.

## B Limitation and Future Work

**Validation at scale.** Our study is limited up to 3B model scales. Since standalone Mamba models often show diminishing returns at scale (Waleffe et al., 2024), a crucial next step is to validate whether the performance advantages of our hybrid model persist with longer training and at larger scales. However, we are optimistic they will, based on our scaling analysis up to the 3B parameter size and prior work suggesting that hybrid models scale more effectively (Waleffe et al., 2024; Jamba Team et al., 2024; Basant et al., 2025).

**Compatibility of advanced primitives.** Our work combined base Transformer and Mamba blocks, whereas recent models incorporate more advanced variants. For instance, some models (De et al. (2024), Ren et al. (2025), Qwen3-Next) employ self-attention variants like local (Beltagy et al., 2020), differential (Ye et al., 2024), and gated attention (Qiu et al., 2025), while others (RWKV-X (Hou et al., 2025), Qwen3-Next, Kimi Linear (Kimi Team, 2025)) use linear attention mechanisms such as RWKV-7(Peng et al., 2025), Gated DeltaNet (Yang et al., 2024), or Kimi Delta Attention (Kimi Team, 2025). A key question is whether our design insights still apply to these newer hybrids and if the advantages of each component are preserved when combined.

**Modality extension.** To achieve superintelligence, models must move beyond language to internalize the laws of physics that govern our world through video modality. This shift to multimodal learning (e.g., video, audio) intensifies the need for architectures that can support tokenization-free processing (Yu et al., 2023; Pagnoni et al., 2024; Assran et al., 2025) and overcome long-context bottlenecks. Consequently, hybrid architectures incorporating SSMs are rapidly gaining traction, making their extension beyond the language domain a critical next step.

## C Comparison with Prior Works

As discussed in Section 3, numerous hybrid architectures have been proposed, generally categorized into inter-layer and intra-layer approaches. To underscore the contributions of our work, we clarify the distinctions from prior studies, specifically highlighting the critical gaps they left unaddressed.

Existing research has predominantly focused on scaling hybrid architectures into foundation models, optimizing data composition and training pipelines (spanning pre-, mid-, and post-training), or introducing novel linear attention variants (e.g., Zamba (Glorioso et al., 2024b), Jamba (Lieber et al., 2024; Jamba Team et al., 2024), Samba (Ren et al., 2024), MiniMax-01 (Li et al., 2025a), Hunyuan-TurboS (Hunyuan Team et al., 2025), RWKV-X (Hou et al., 2025), Nemotron Nano 2 (Basant et al., 2025), Hymba (Dong et al., 2024), and Falcon-H1 (Zuo et al., 2025)). While these works successfully demonstrate reasoning capabilities and long-context performance through scaling, the rationale behind their specific architectural choices is still unclear. Often, only the final architectures are presented without fully disclosing the *recipes* or design processes that led to those decisions.

Although there are limited exceptions, they often rely on verbal explanations or restricted ablations. In inter-layer hybrids, discussions on block ratios or positioning are sparse; for instance, Jamba (Lieber et al., 2024; Jamba Team et al., 2024), Waleffe et al. (2024), RWKV-X (Hou et al., 2025), and Kimi-Linear (Kimi Team, 2025) compare a limited set of block ratios, while Waleffe et al. (2024) and Samba (Ren et al., 2024) investigate the positioning of FFN or self-attention blocks in specific configurations. Similarly, in the intra-layer domain, experimental evidence regarding design variants remains limited. Hymba (Dong et al., 2024), for example, shares results for only a single concatenation variant, and Falcon-H1 (Zuo et al., 2025) focuses its deep ablations primarily on the Mamba component itself rather than the hybridization strategy (although both works include ablation studies regarding dimension ratios).

Furthermore, a rigorous comparison regarding model quality and efficiency between inter-layer and intra-layer approaches has been absent. Even within intra-hybridization strategies, identifying the best-performing variants or comparing them against alternative architectural configurations employing the same modules (e.g., differential architectures (Ye et al., 2024; Schneider et al., 2025)) has remained an open question.

We believe our work bridges this gap. We distinguish our contributions by:

- Openly sharing architectural insights on optimal design choices for each hybridization strategy, grounded in a comprehensive study.
- Conducting an in-depth comparison of the trade-offs between inter-layer and intra-layer hybrids—two streams that have traditionally been researched in parallel.

We believe these analyses will serve as a valuable guideline for the community, facilitating the development of more robust hybrid architectures.

## D Details for computational and memory costs comparison

**FLOPs per token comparison.** Most parameters are linear weight matrices, so total FLOPs can be estimated by multiplying the parameter count by 6 (accounting for addition, multiplication, and forward / backward passes) For Transformers, attention further adds FLOPs from query-key dot products and value scaling, totaling $12N_L d_{\text{model}}(L_{\text{ctx}}+1)/2$, considering only causal attention and excluding Flash-Attention overhead (Dao et al., 2022). In SWA, FLOPs depend on the number of activated tokens based on window and sink sizes. Mamba, on the other hand, introduces additional per-token FLOPs from its work-efficient parallel scan algorithm (Blelloch, 1990; Smith et al., 2022), calculated as $3(9d_{\text{ssm}}d_{\text{state}} + 2d_{ssm})$. Thus, FLOPs per sample scale quadratically with sequence length for Transformer, while Mamba scales linearly. For a 1B model with 8K context, Mamba uses about 18% fewer FLOPs than a Transformer.

**Parameter counts comparison.** The attention in Transformer block uses four projection weights—query, key, value, and output. Variants like GQA reduce parameter count by decreasing the number of key and value heads, shrinking the corresponding projection matrices. In contrast, Mamba featurizes the input for state space parameters using several projection weights and 1D convolution layers, projecting the hidden dimension to twice its original size. It also includes time-invariant parameters $\mathbf{A}$ and $\mathbf{D}$, as well as gating and output projection weights. As a result, when comparing only the attention components (excluding FFN), Mamba's parameter count is approximately 2.5 times higher than that of a Transformer block in a 1B model (e.g., 25M vs. 10M parameters per block).

**Cache size comparison.** Cache size is a major inference bottleneck due to the GPU memory hierarchy, increasing HBM–SRAM communication. In Transformers, the cache consists of key and value states. The cache size is $4N_L N_{\text{kv}} d_{\text{head}} L_{\text{ctx}}$, where 4 accounts for key, value, and 2 bytes per `bfloat16` element. Meanwhile, Mamba maintains two types of caches: hidden states for convolution layer and memory states compressed into a finite space. Its total cache size is $2N_L(N_{\text{conv}}(2d_{\text{state}} + d_{\text{ssm}}) + d_{\text{ssm}}d_{\text{state}})$, with `bfloat16` precision. Notably, Mamba's cache size is independent of sequence length, so its efficiency benefits increase with longer contexts. For a 1B model with 8K context, Mamba's cache footprint is about 95% smaller than a Transformer's (e.g., 256 MiB vs. 13.4 MiB).

## E Details for Intra-layer Hybrid Architectures

Since intra-layer hybrid architectures integrate two distinct modules within a single layer, they offer a vast design space for optimization. We explore this flexibility through five primary axes of variation: output projection, scaling factors, normalization, fusion operation, and shared input values. These design choices can be formulated as the following generalized equation:

$$f(\mathbf{X}) = \mathbf{W}_{\text{out}} \left( \alpha_{ssm} \cdot \text{Norm}\left(\mathcal{M}_{\text{ssm}}(\mathbf{X})\right) \oplus \alpha_{attn} \cdot \text{Norm}\left(\mathcal{M}_{\text{attn}}(\mathbf{X})\right) \right),$$

$$\text{where} \quad \mathcal{M}_{\text{ssm}}(\mathbf{X}) = \mathbf{G} \odot \left( \mathbf{C} \prod \exp(\mathbf{A}\Delta)\mathbf{B}\Delta \right) \mathbf{W}^{\text{ssm}}\mathbf{X},$$

$$\mathcal{M}_{\text{attn}}(\mathbf{X}) = \text{softmax}\left( \mathbf{Q}\mathbf{K}^{\top}/\sqrt{d_k} \right) \mathbf{W}^{V}\mathbf{X}.$$

Here, $\mathcal{M}_{\text{attn}}$ and $\mathcal{M}_{\text{ssm}}$ denote the global attention and SSM modules, respectively. In the SSM formulation, $\mathbf{G}$ represents the gating branch, and the middle term denotes the discretized convolution kernel derived from the state-space parameters (indices are omitted for simplicity). $\mathbf{W}^{\text{ssm}}$ includes a projection layer and a 1D convolution layer. The colored terms highlight the five configurable axes. Table 6 summarizes the specific variants explored across each design axis. Our search space is comprehensive, encompassing strategies adopted in prior studies as well as previously unexplored configurations to ensure robust validation.

Table 6: **Overview of the architectural design space for intra-layer hybrid architectures.** We explore various design choices across five distinct axes.

| Design Axis | Explored Variants |
|---|---|
| **Output Projection** ($W_{\text{out}}$) | (1) **Unified**: Single projection shared after fusion.<br>(2) **Separate**: Individual projections for each module before fusion. |
| **Normalization** (Norm) | (1) **Module-wise**: Applied individually to each module output.<br>(2) **None**: Raw module outputs are used without normalization. |
| **Scaling Strategy** ($\alpha$) | (1) **Element-wise**: Per-module learnable vector ($\alpha \in \mathbb{R}^d$).<br>(2) **Asymmetric**: Learnable scalar only for second module ($\alpha \in \mathbb{R}$).<br>(3) **Identity**: No scaling applied ($\alpha = 1$). |
| **Fusion Operation** ($\oplus$) | (1) **Addition**: Standard summation ($\mathbf{y} = \mathcal{M}_S + \mathcal{M}_A$)<br>(2) **Subtraction**: Differential interaction ($\mathbf{y} = \mathcal{M}_S - \mathcal{M}_A$)<br>(3) **Concatenation**: Channel-wise concat ($\mathbf{y} = [\mathcal{M}_S; \mathcal{M}_A]$) |
| **Value Sharing** | (1) **None**: Independent projections from layer input $\mathbf{X}$.<br>(2) **Value State**: Mamba reuses Attention's value ($\mathbf{V} = \mathbf{W}^{V}\mathbf{X}$). |

While most axes are self-explanatory, the *Value Sharing* strategy needs further elaboration, particularly regarding its inspiration from differential architectures (Ye et al., 2024). The Differential Transformer is designed to cancel noise in attention scores by subtracting the attention maps derived from two distinct head groups (i.e., $\texttt{softmax}(\mathbf{Q}_1\mathbf{K}_1^{\top}) - \texttt{softmax}(\mathbf{Q}_2\mathbf{K}_2^{\top})$). Crucially, to preserve model expressivity, it applies the differential attention map to a Value state ($\mathbf{V} = \mathbf{W}^{\mathbf{V}}\mathbf{X}$) that retains the original full head dimension. Adopting this insight, we incorporate the value dimension expansion technique into our $\mathcal{M}_{\text{attn}}$ module.

Furthermore, we extend this logic to the hybrid domain: we hypothesize that sharing the Value states can induce a denoising or synergistic fusion effect between the global attention and Mamba's implicit attention mechanism. Consequently, we introduce a variant where the Mamba module utilizes the Attention's value projection directly (i.e., using $\mathbf{W}^{V}\mathbf{X}$ as input instead of a separate projection $\mathbf{W}^{\text{ssm}}\mathbf{X}$). Finally, unlike previous works, we rigorously benchmark our models against homogeneous differential architectures (Ye et al., 2024; Schneider et al., 2025) employing identical module configurations to validate the distinct advantages of hybridization.

## F Details for Experimental Setup

**Model architecture details.** Each model is constructed based on the Llama-based Transformer (Llama Team, 2024) and the Mamba architectures (Dao & Gu, 2024). We primarily follow to the configurations of Llama 3.2 and Mamba 2 across various model scales. Table 7 provides a summary of the detailed architectures for both the Transformer and Mamba models, which serve as the foundational computational primitives for our hybrid architecture variants.

Table 7: **Comparison of architectural configurations for Transformer and Mamba models across different scales.** For clarity, we separately detail the specific configurations for Attention and SSM components. When constructing the inter- and intra-layer hybrid architectures, we refer to these configurations as the base computational primitives.

| | | Base Configuration | | | | | | | Attention | | SSM | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Models** | **Sizes** | N-emb | Emb | Vocab | $N_L$ | $d_{\text{model}}$ | $d_{\text{ffn}}$ | $N_{\text{head}}$ | $N_{kv}$ | $d_{\text{head}}$ | $d_{\text{ssm}}$ | $d_{\text{head}}$ | $d_{\text{state}}$ | $N_{\text{conv}}$ |
| Llama | 100M | 0.10B | 0.13B | 128K | 8 | 1024 | 3072 | 16 | 4 | 64 | - | - | - | - |
| | 350M | 0.35B | 0.20B | 128K | 14 | 1536 | 4096 | 24 | 8 | 64 | - | - | - | - |
| | 1B | 0.97B | 0.26B | 128K | 16 | 2048 | 8192 | 32 | 8 | 64 | - | - | | |
| | 3B | 2.78B | 0.39B | 128K | 28 | 3072 | 8192 | 32 | 8 | 96 | - | - | - | - |
| Mamba | 100M | 0.10B | 0.13B | 128K | 6 | 1024 | 3072 | 16 | - | - | 2048 | 128 | 128 | 4 |
| | 350M | 0.37B | 0.20B | 128K | 11 | 1536 | 4096 | 24 | - | - | 3072 | 128 | 128 | 4 |
| | 1B | 0.98B | 0.26B | 128K | 13 | 2048 | 8192 | 32 | - | - | 4096 | 128 | 128 | 4 |
| | 3B | 2.80B | 0.39B | 128K | 21 | 3072 | 8192 | 32 | - | - | 6144 | 192 | 256 | 4 |

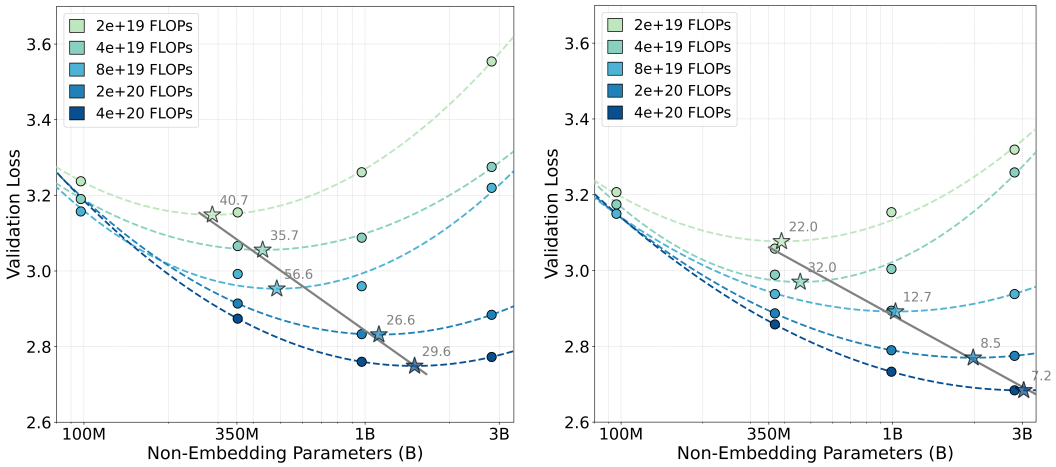
**Training settings.** We conducted experiments using TorchTitan (Liang et al., 2024), a PyTorch-native platform designed for large-scale training of LLMs. We pretrain different hybrid models variants from scratch on DCLM-Baseline pretraining dataset (Li et al., 2024), which contains 4 trillion tokens from 3 billion documents. We pretrained on randomly sampled 60 billion tokens using 8 H100 or H200 GPUs by default. We packed the corpus using the Llama 3.2 tokenizer (Llama Team, 2024), which has a vocabulary size of 128K, with a context length of 8K tokens. We enable the model to attend to all previous tokens, not just those within the same document, since the `eos` and `bos` tokens are used to distinguish document boundaries. We employed only FSDP (Zhao et al., 2023) with a degree equal to the number of GPUs, and activation checkpointing to reduce memory footprint. We used a batch size of 2 million tokens and utilized a trapezoid learning rate scheduler (Xing et al., 2018) consisting of warmup (about 25%), stable, and cool down (about 20%) phases. The AdamW optimizer (Loshchilov & Hutter, 2017) and gradient clipping were used for all experiments. For the different model scales—100M, 350M, 1B, and 3B parameters—the learning rates were set to 6e-3, 3e-3, 6e-4, and 3e-4, respectively. We utilized `torch.compile` (Ansel et al., 2024) to accelerate training.

**Evaluation settings.** To assess model performance, we evaluated language modeling performance on the validation set of DCLM-Baseline (approximately 8,000 samples) (Li et al., 2024) and the PG19 dataset (approximately 150 samples) (Rae et al., 2019b). Additionally, following the settings of prior work (Gemini Team et al., 2024; Ho et al., 2024), we conducted evaluations on the Needle-In-a-Haystack long-context benchmark (Kamradt, 2023; Kuratov et al., 2024). We further evaluated few-shot accuracy on five benchmarks using the Language Model Evaluation Harness (Gao et al., 2024): LAMBADA (LD), HellaSwag (HS), PIQA (PQ), ARC (Easy and Challenge), and OpenBookQA (OB). For all the few-shot datasets except LAMBADA, accuracy was normalized by the byte length of the target string. We adhered to the standard number of shots for each dataset. All evaluation experiments were performed on a single H100 or H200 GPU.

## G Expanded Results of Scaling Laws Experiment

We conduct a scaling law analysis by training four model scales (100M, 350M, 1B, and 3B) across five compute budgets (ranging from 2e19 to 4e20 FLOPs). We efficiently conduct all budget experiments with a Trapezoid learning rate scheduler and reusing intermediate checkpoints from the training run with the largest budget. The warmup stage is therefore defined as one-quarter of the step count of the smallest budget. The learning rates used for the 100M, 350M, 1B, and 3B models are 6e-3, 3e-3, 6e-4, and 3e-4, respectively. We use 0.5M tokens/step, and all other settings, such as the context length of 8K, remain identical to the main experiment. The models are trained on the DCLM training set and evaluated on 8,000 validation samples.

Based on this, we derive the optimal compute-scaling lines, as shown in the right panel of Table 3. The IsoFLOPs analysis results for each architecture are visualized in Figure G, with detailed numerical results summarized in Tables 8 and 9.




(a) Transformer (b) Mamba

(c) Inter-layer hybrid (d) Inter-layer hybrid

Figure 5: **Visualization of scaling law results for each architecture.** We apply a quadratic fit to the results at each compute budget. Star markers denote the optimal parameter size achieving the best quality for each budget. The numbers on the optimal scaling frontier represent the token-to-parameter ratio.

Table 8: **Detailed scaling law experiments for Transformer and Mamba models.** We pre-train four scales of models, up to 3B parameters, from scratch across five Training FLOPs budgets and evaluated their negative log-likelihood (NLL). The models are trained on the DCLM training set and evaluated on 8,000 validation samples.

| | | Base Config | | | | | | | | | | Compute | | | NLL (↓) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Models** | **N-emb** | $N_{\text{attn}}$ | $N_{\text{ssm}}$ | $N_{\text{hyb}}$ | $d_{\text{model}}$ | $d_{\text{ffn}}$ | $N_{\text{head}}$ | $N_{\text{kv}}$ | $d_{\text{ssm}}$ | $d_{\text{state}}$ | $N_{\text{conv}}$ | FLOPs | Tok | Steps | DCLM |
| Llama | 0.10B | 8 | - | - | 1024 | 3072 | 16 | 4 | - | - | - | 2e19 | 20B | 39K | 3.237 |
| | | | | | | | | | | | | 4e19 | 41B | 78K | 3.190 |
| | | | | | | | | | | | | 8e19 | 82B | 155K | 3.157 |
| | 0.35B | 14 | - | - | 1536 | 4096 | 24 | 8 | - | - | - | 2e19 | 6B | 12K | 3.155 |
| | | | | | | | | | | | | 4e19 | 13B | 24K | 3.066 |
| | | | | | | | | | | | | 8e19 | 25B | 48K | 2.992 |
| | | | | | | | | | | | | 2e20 | 63B | 120K | 2.914 |
| | | | | | | | | | | | | 4e20 | 126B | 241K | 2.874 |
| | 0.97B | 16 | - | - | 2048 | 8192 | 32 | 8 | - | - | - | 2e19 | 3B | 5K | 3.261 |
| | | | | | | | | | | | | 4e19 | 5B | 10K | 3.088 |
| | | | | | | | | | | | | 8e19 | 11B | 20K | 2.960 |
| | | | | | | | | | | | | 2e20 | 27B | 51K | 2.833 |
| | | | | | | | | | | | | 4e20 | 54B | 102K | 2.760 |
| | 2.82B | 28 | - | - | 3072 | 8192 | 32 | 8 | - | - | - | 2e19 | 1B | 2K | 3.554 |
| | | | | | | | | | | | | 4e19 | 2B | 4K | 3.275 |
| | | | | | | | | | | | | 8e19 | 4B | 7K | 3.212 |
| | | | | | | | | | | | | 2e20 | 10B | 18K | 2.884 |
| | | | | | | | | | | | | 4e20 | 19B | 37K | 2.773 |
| Mamba | 0.10B | - | 6 | - | 1024 | 3072 | 16 | - | 2048 | 128 | 4 | 2e19 | 27B | 51K | 3.173 |
| | | | | | | | | | | | | 4e19 | 54B | 102K | 3.134 |
| | 0.34B | - | 11 | - | 1536 | 4096 | 24 | - | 3072 | 128 | 4 | 2e19 | 8B | 16K | 3.050 |
| | | | | | | | | | | | | 4e19 | 17B | 32K | 2.983 |
| | | | | | | | | | | | | 8e19 | 34B | 65K | 2.933 |
| | | | | | | | | | | | | 2e20 | 85B | 162K | 2.884 |
| | | | | | | | | | | | | 4e20 | 170B | 324K | 2.855 |
| | 0.99B | - | 13 | - | 2048 | 8192 | 32 | - | 4096 | 128 | 4 | 2e19 | 3B | 6K | 3.159 |
| | | | | | | | | | | | | 4e19 | 6B | 12K | 3.003 |
| | | | | | | | | | | | | 8e19 | 12B | 24K | 2.885 |
| | | | | | | | | | | | | 2e20 | 31B | 59K | 2.772 |
| | | | | | | | | | | | | 4e20 | 62B | 119K | 2.712 |
| | 2.81B | - | 21 | - | 3072 | 8192 | 32 | - | 6144 | 256 | 4 | 2e19 | 1B | 2K | 3.367 |
| | | | | | | | | | | | | 4e19 | 2B | 4K | 3.133 |
| | | | | | | | | | | | | 8e19 | 4B | 8K | 2.962 |
| | | | | | | | | | | | | 2e20 | 11B | 20K | 2.790 |
| | | | | | | | | | | | | 4e20 | 21B | 41K | 2.691 |

Table 9: **Detailed scaling law experiments for Inter- and Intra-layer hybrid models.** Both hybrid models use a 1:5 block ratio. The Inter-hybrid model adopts a structure where the global attention layers are uniformly distributed throughout the depth (see Section 4.5). The Intra-hybrid model follows the final structure selected within the block (see Section 4.6). The global attention component fundamentally follows the structure of the Differential Transformer (Ye et al., 2024). In this setup, the query and key have a half-size dimension ($d_{\text{head}}/2$) corresponding to the half the number of assigned heads, while the value retains the full $d_{\text{model}}$. For the Mamba component, a separate output projection is used to restore the $d_{\text{model}}$ from the $d_{\text{ssm}}$ before the two modules are fused. To ensure a fair parameter count, the 1B Intra-layer hybrid model specifically uses a query and key dimension of 1152 and a $d_{\text{ssm}}$ of 2304.

| | | Base Config | | | | | | | | | | Compute | | | NLL (↓) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Models** | **N-emb** | $N_{\text{attn}}$ | $N_{\text{ssm}}$ | $N_{\text{hyb}}$ | $d_{\text{model}}$ | $d_{\text{ffn}}$ | $N_{\text{head}}$ | $N_{\text{kv}}$ | $d_{\text{ssm}}$ | $d_{\text{state}}$ | $N_{\text{conv}}$ | FLOPs | Tok | Steps | DCLM |
| Inter-H | 0.09B | 5 | 1 | - | 1024 | 3072 | 16 | 4 | 2048 | 128 | 4 | 2e19 | 31B | 60K | 3.192 |
| | | | | | | | | | | | | 4e19 | 63B | 119K | 3.159 |
| | | | | | | | | | | | | 8e19 | 125B | 239K | 3.136 |
| | 0.35B | 2 | 9 | - | 1536 | 4096 | 24 | 8 | 3072 | 128 | 4 | 2e19 | 8B | 16K | 3.037 |
| | | | | | | | | | | | | 4e19 | 17B | 32K | 2.972 |
| | | | | | | | | | | | | 8e19 | 34B | 65K | 2.919 |
| | | | | | | | | | | | | 2e20 | 85B | 162K | 2.870 |
| | | | | | | | | | | | | 4e20 | 170B | 324K | 2.842 |
| | 0.96B | 2 | 11 | - | 2048 | 8192 | 32 | 8 | 4096 | 128 | 4 | 2e19 | 3B | 6K | 3.128 |
| | | | | | | | | | | | | 4e19 | 7B | 12K | 2.977 |
| | | | | | | | | | | | | 8e19 | 13B | 25K | 2.869 |
| | | | | | | | | | | | | 2e20 | 33B | 62K | 2.769 |
| | | | | | | | | | | | | 4e20 | 65B | 125K | 2.714 |
| | 2.81B | 4 | 18 | - | 3072 | 8192 | 32 | 8 | 6144 | 256 | 4 | 2e19 | 1B | 2K | 3.403 |
| | | | | | | | | | | | | 4e19 | 2B | 4K | 3.147 |
| | | | | | | | | | | | | 8e19 | 4B | 8K | 2.962 |
| | | | | | | | | | | | | 2e20 | 11B | 21K | 2.777 |
| | | | | | | | | | | | | 4e20 | 22B | 42K | 2.672 |
| Intra-H | 0.10B | - | 5 | 1 | 1024 | 3072 | 8+8 | 2 | 1024 | 128 | 4 | 2e19 | 31B | 59K | 3.186 |
| | | | | | | | | | | | | 4e19 | 62B | 118K | 3.155 |
| | | | | | | | | | | | | 8e19 | 124B | 236K | 3.131 |
| | 0.36B | - | 9 | 2 | 1536 | 4096 | 16+16 | 4 | 1536 | 128 | 4 | 2e19 | 8B | 16K | 3.037 |
| | | | | | | | | | | | | 4e19 | 17B | 32K | 2.969 |
| | | | | | | | | | | | | 8e19 | 33B | 64K | 2.917 |
| | | | | | | | | | | | | 2e20 | 83B | 160K | 2.867 |
| | | | | | | | | | | | | 4e20 | 167B | 318K | 2.838 |
| | 0.98B | - | 11 | 2 | 2048 | 8192 | 16+16 | 4 | 2304 | 128 | 4 | 2e19 | 3B | 6K | 3.123 |
| | | | | | | | | | | | | 4e19 | 6B | 12K | 2.977 |
| | | | | | | | | | | | | 8e19 | 13B | 24K | 2.868 |
| | | | | | | | | | | | | 2e20 | 32B | 61K | 2.765 |
| | | | | | | | | | | | | 4e20 | 64B | 122K | 2.709 |
| | 2.89B | - | 18 | 4 | 3072 | 8192 | 16+16 | 4 | 3072 | 256 | 4 | 2e19 | 1B | 2K | 3.425 |
| | | | | | | | | | | | | 4e19 | 2B | 4K | 3.156 |
| | | | | | | | | | | | | 8e19 | 4B | 8K | 2.968 |
| | | | | | | | | | | | | 2e20 | 11B | 20K | 2.782 |
| | | | | | | | | | | | | 4e20 | 21B | 41K | 2.676 |

# H Expanded Results of Intra-layer Hybridization Ablation

## H.1 Insights for Architectural Variants

To thoroughly analyze the design space of intra-layer hybrid architectures, we conduct a comprehensive evaluation at both the 350M and 1B parameter scales. We specifically examine six pivotal design axes: primitive types, normalization layers, scaling formulations, fusion operations, value sharing strategies, and output projection configurations. Tables 10 and 11 present the results of our ablation study. The best-performing candidates (considering both model quality and efficiency) are highlighted in gray. We finally select the `Diff` fusion case as the final structure due to its systemic simplicity, which includes favorable compatibility with expert parallelism.

Table 10: **Detailed ablation results related to architectural variants for intra-layer hybridization at 1B scale.** All models are pretrained on 60B tokens from the DCLM dataset. We replace all depths to intra-hybrid blocks (1:0 ratio), which consist of global attention (GQA) and Mamba primitives in a head-wise splitting manner. † denotes our re-implementation of prior methods, incorporating value dimension expansion (Ye et al., 2024) for GQA. We highlight the final best-performing candidates in gray.

| | **Base Config** | | **Design Choices** | | | | | | **NLL (↓)** | | **Few-shot Accuracy (↑)** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Models** | N-emb | $N_L$ | Module | Norm | Scalar | Fusion | Value | Out | DCLM | PG19 | LD | HS | PQ | ARC | OB | Avg |
| Llama | 0.97B | 16 | - | - | - | - | - | - | 2.750 | 2.875 | **56.1** | 55.2 | 72.8 | 43.2 | 32.7 | 52.0 |
| Mamba | 0.99B | 13 | - | - | - | - | - | - | 2.758 | 2.891 | 53.7 | 55.1 | 73.8 | 43.9 | 35.2 | 52.3 |
| Diff-T | 0.97B | 16 | T/T | - | Diff-T | Diff | Joint | Joint | 2.727 | 2.848 | 58.2 | 57.1 | 72.9 | 44.3 | 35.4 | 53.6 |
| Diff-M | 0.99B | 13 | M/M | - | Diff-M | Diff | Sep | Sep | 2.759 | 2.892 | 55.0 | 56.0 | 73.7 | 45.2 | 35.0 | 53.0 |
| ∟Variants | 0.97B | 16 | T/T | - | Diff-T | Add | Joint | Joint | 2.733 | 2.856 | 56.0 | 56.5 | 73.4 | 45.5 | 36.0 | 53.5 |
| | 0.96B | 15 | T/M | - | Diff-T | Diff | Joint | Joint | 2.733 | 2.854 | 56.3 | 56.5 | 74.3 | 46.4 | 36.8 | 54.1 |
| | 0.96B | 15 | T/M | Group | Diff-T | Diff | Joint | Joint | 2.735 | 2.855 | 55.1 | 56.3 | 73.3 | 45.0 | 35.2 | 53.0 |
| | 0.94B | 14 | T/M | - | Diff-T | Diff | Sep | Joint | 2.731 | 2.852 | 56.7 | 57.0 | 74.1 | 46.1 | 36.8 | 54.1 |
| | 0.94B | 14 | T/M | Group | Diff-T | Diff | Sep | Joint | 2.728 | 2.852 | 57.4 | 55.9 | 73.1 | 46.5 | 35.4 | 53.6 |
| | 0.95B | 13 | T/M | Group | Diff-T | Diff | Sep | Sep | 2.719 | 2.838 | 58.3 | 57.2 | 73.2 | 45.9 | 35.4 | 54.0 |
| Hymba† | 0.94B | 14 | T/M | Group | Scale | Add | Sep | Joint | 2.726 | 2.846 | 56.2 | 56.8 | 73.2 | 45.9 | 36.2 | 53.7 |
| ∟Variants | 0.94B | 14 | T/M | - | - | Add | Sep | Joint | 2.738 | 2.861 | 58.2 | 56.5 | **74.2** | 46.3 | 37.0 | 54.4 |
| | 0.94B | 14 | T/M | Group | - | Add | Sep | Joint | 2.721 | 2.840 | 59.0 | 57.2 | 73.6 | 45.8 | 37.0 | 54.5 |
| | 0.94B | 14 | T/M | - | Scale | Add | Sep | Joint | 2.740 | 2.865 | 57.7 | 56.2 | 73.8 | 45.6 | 36.0 | 53.9 |
| | 0.96B | 15 | T/M | Group | Scale | Add | Joint | Joint | 2.729 | 2.850 | 56.9 | 57.2 | 73.9 | 45.8 | 36.6 | 54.1 |
| | 0.95B | 13 | T/M | - | - | Add | Sep | Sep | 2.720 | 2.842 | 58.8 | 57.3 | 73.0 | 45.5 | 35.8 | 54.1 |
| | 0.95B | 13 | T/M | Group | - | Add | Sep | Sep | **2.714** | 2.834 | 58.4 | 57.6 | 74.7 | 46.3 | 35.6 | 54.5 |
| | 0.95B | 13 | T/M | Group | Scale | Add | Sep | Sep | 2.720 | 2.841 | 57.7 | 56.4 | 72.9 | 45.2 | 35.2 | 53.5 |
| | 0.95B | 13 | T/M | - | - | Diff | Sep | Sep | 2.715 | 2.836 | 59.7 | 57.5 | 73.7 | 46.4 | 35.8 | 54.6 |
| | 0.95B | 13 | T/M | Group | - | Diff | Sep | Sep | 2.712 | 2.831 | **59.8** | **57.9** | 73.1 | **47.3** | 36.2 | **54.9** |
| | 0.95B | 13 | T/M | Group | Scale | Diff | Sep | Sep | 2.727 | 2.846 | 59.3 | 56.5 | 73.0 | 44.8 | 37.2 | 54.2 |
| | 0.94B | 14 | T/M | Group | - | Diff | Sep | Joint | 2.721 | 2.842 | 58.3 | 57.0 | 73.6 | 45.1 | 35.8 | 54.0 |
| Falcon-H1† | 0.95B | 13 | T/M | - | - | Conc | Sep | Joint | 2.720 | 2.840 | 56.6 | 56.8 | 73.0 | 46.3 | 35.2 | 53.6 |
| ∟Variants | 0.95B | 13 | T/M | Group | - | Conc | Sep | Joint | 2.715 | **2.833** | 59.0 | 57.7 | 73.7 | 45.8 | **37.6** | 54.8 |
| | 0.95B | 13 | T/M | Group | Scale | Conc | Sep | Joint | 2.724 | 2.842 | 57.8 | 57.0 | 73.6 | 47.2 | 36.0 | 54.3 |

Table 11: **Detailed ablation results related to architectural variants for intra-layer hybridization at 350M scale.** All models are pretrained on 60B tokens from the DCLM dataset. We replace all depths to intra-hybrid blocks (1:0 ratio), which consist of global attention (GQA) and Mamba primitives in a head-wise splitting manner. † denotes our re-implementation of prior methods, incorporating value dimension expansion (Ye et al., 2024) for GQA. We highlight the final best-performing candidates in gray.

| | Base Config | | Design Choices | | | | | | NLL (↓) | | Few-shot Accuracy (↑) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Models** | N-emb | $N_L$ | Module | Norm | Scalar | Fusion | Value | Out | DCLM | PG19 | LD | HS | PQ | ARC | OB | Avg |
| Llama | 0.35B | 14 | - | - | - | - | - | - | 2.882 | 3.015 | 51.5 | 48.3 | 70.4 | 40.4 | 33.0 | 48.7 |
| Mamba | 0.37B | 11 | - | - | - | - | - | - | 2.880 | 3.024 | 49.4 | 49.1 | 71.3 | 41.1 | 32.6 | 48.7 |
| Diff-T | 0.35B | 14 | T/T | - | Diff-T | Diff | Joint | Joint | 2.858 | 2.991 | 56.1 | 50.0 | 70.7 | 40.3 | 32.8 | 50.0 |
| Diff-M | 0.37B | 11 | M/M | - | Diff-M | Diff | Sep | Sep | 2.879 | 3.024 | 50.1 | 49.4 | **71.9** | 41.2 | 34.2 | 49.4 |
| ∟Variants | 0.35B | 14 | T/T | - | Diff-T | Add | Joint | Joint | 2.865 | 3.001 | 51.9 | 49.2 | 71.4 | 41.4 | 32.8 | 49.3 |
| | 0.35B | 13 | T/M | - | Diff-T | Diff | Joint | Joint | 2.862 | 2.994 | 52.8 | 49.5 | 70.5 | **42.6** | 33.4 | 49.7 |
| | 0.35B | 13 | T/M | Group | Diff-T | Diff | Joint | Joint | 2.859 | 2.990 | **50.6** | 49.5 | 71.8 | 40.2 | 32.4 | 48.9 |
| | 0.34B | 12 | T/M | - | Diff-T | Diff | Sep | Joint | 2.864 | 3.002 | 52.4 | 49.9 | 69.9 | 40.2 | 31.6 | 48.8 |
| | 0.34B | 12 | T/M | Group | Diff-T | Diff | Sep | Joint | 2.856 | 2.989 | 50.8 | 50.1 | 71.6 | 41.5 | 33.0 | 49.4 |
| | 0.36B | 11 | T/M | Group | Diff-T | Diff | Sep | Sep | 2.853 | 2.987 | 54.9 | 50.2 | 71.0 | 41.1 | 34.4 | 50.3 |
| Hymba† | 0.34B | 12 | T/M | Group | Scale | Add | Sep | Joint | 2.861 | 2.995 | 51.5 | 49.4 | 71.4 | 42.2 | 33.8 | 49.7 |
| ∟Variants | 0.34B | 12 | T/M | - | - | Add | Sep | Joint | 2.870 | 3.006 | 52.6 | 49.2 | 71.1 | 39.5 | 32.8 | 49.0 |
| | 0.34B | 12 | T/M | Group | - | Add | Sep | Joint | 2.857 | 2.989 | 54.0 | 50.0 | 71.0 | 41.5 | 34.2 | 50.1 |
| | 0.34B | 12 | T/M | - | Scale | Add | Sep | Joint | 2.872 | 3.010 | 54.1 | 49.3 | 71.3 | 40.5 | 33.0 | 49.6 |
| | 0.35B | 13 | T/M | Group | Scale | Add | Joint | Joint | 2.861 | 2.993 | 52.4 | 49.4 | 71.0 | 41.3 | 32.8 | 49.4 |
| | 0.36B | 11 | T/M | - | - | Add | Sep | Sep | 2.851 | 2.985 | 51.9 | 50.0 | 70.5 | 40.0 | 33.0 | 49.1 |
| | 0.36B | 11 | T/M | Group | - | Add | Sep | Sep | 2.849 | 2.985 | 51.6 | 50.6 | 71.6 | 40.8 | 33.8 | 49.7 |
| | 0.36B | 11 | T/M | Group | Scale | Add | Sep | Sep | 2.849 | 2.987 | 51.7 | 50.4 | 70.8 | 40.8 | 32.8 | 49.3 |
| | 0.36B | 11 | T/M | - | - | Diff | Sep | Sep | 2.850 | 2.983 | 53.6 | 50.5 | 71.4 | 41.2 | 34.0 | 50.1 |
| | 0.36B | 11 | T/M | Group | - | Diff | Sep | Sep | 2.847 | 2.983 | 51.5 | 50.0 | 71.5 | 41.1 | 32.8 | 49.4 |
| | 0.36B | 11 | T/M | Group | Scale | Diff | Sep | Sep | 2.851 | 2.987 | 54.0 | 49.7 | 71.2 | 41.5 | 34.0 | 50.1 |
| | 0.34B | 12 | T/M | Group | - | Diff | Sep | Joint | 2.858 | 2.990 | 54.3 | 49.9 | 70.9 | 42.4 | 33.6 | 50.2 |
| Falcon-H1† | 0.36B | 11 | T/M | - | - | Conc | Sep | Joint | 2.855 | 2.989 | 54.2 | 50.3 | 70.5 | 42.2 | 32.6 | 50.0 |
| ∟Variants | 0.36B | 11 | T/M | Group | - | Conc | Sep | Joint | **2.846** | **2.978** | 55.2 | **50.6** | 71.1 | 41.7 | **34.4** | **50.6** |
| | 0.36B | 11 | T/M | Group | Scale | Conc | Sep | Joint | 2.851 | 2.983 | 53.1 | 49.9 | 72.3 | 41.7 | 34.8 | 50.3 |

Based on these extensive experiments, we observe the following:

- **Modules**: The combination of global attention and Mamba significantly outperforms both pure homogeneous models and differential models (Ye et al., 2024; Schneider et al., 2025).
- **Normalization**: Applying `Group` normalization to the output of each module generally improves performance. This appears to stabilize optimization by addressing the significant scale discrepancy ($> 10\times$) between the two modules.
- **Scaling strategies**: While various strategies exist for learning scaling factors, specific methods like `Diff-T` from Differential Transformer (Ye et al., 2024) or the `Group` normalization scheme used in Hymba (Dong et al., 2024) actually lead to performance degradation.
- **Fusion operations**: The choice of fusion operation shows negligible impact on performance, though it closely interacts with value sharing and output projection configurations. We hypothesize that because Mamba's implicit attention scores can inherently be negative, specific fusion mechanisms are not strictly necessary.
- **Value sharing**: Sharing values between modules causes only a minor performance drop. However, under a fixed parameter budget, this strategy results in an increase in the number of layers ($N_L$), negatively impacting efficiency.
- **Output projections**: Maintaining and computing separate output projections for each module (prior to fusion) proves crucial for achieving significant performance gains.

Given the high variance in few-shot accuracy (particularly at smaller scales like 350M), we identify the final candidates (highlighted in gray) based on the NLL metric and the number of layers ($N_L$), which directly impacts efficiency. From these candidates, we ultimately adopt the case of subtracting fusion method for all subsequent experiments.

# I Analysis of Component Contributions in Hybrid Architectures

We analyze the contribution of each module to determine the optimal activation patterns for the Transformer and Mamba components within a hybrid model. To this end, we pretrain a 350M-parameter intra-layer hybrid architecture from scratch, where every layer consists of an intra-hybrid block. In this setup, we divide the heads evenly, normalize the outputs of each module using Group normalization, and combine them via a weighted sum scaled by $\alpha$ and $1 - \alpha$ before the output projection. Here, the value of $\alpha$ serves as a quantitative measure of each module's contribution to next-token prediction.

Using a checkpoint trained on approximately 4B tokens, we investigate how $\alpha$ values are determined on the DCLM validation set. Figure 6 illustrates the average coefficients across each layer for about 100 samples. Consistent with our ablation studies, the model assigns a higher weight to the Mamba component in earlier layers—particularly the first layer—to optimize performance, while assigning higher weights to the Transformer component in the middle layers (specifically layers 6 and 7). This indicates that the global attention pattern at earlier depths—which tends to uniformly attend to all previous contexts—is not optimal in this hybrid setting, while at intermediate depths, the global attention mechanism appears to compensate for Mamba's limited recall capability.

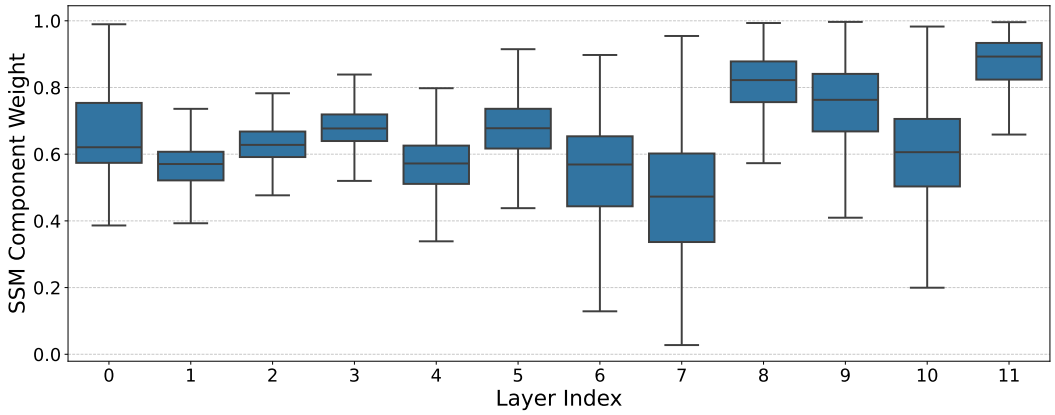



Figure 6: **Box plot of layer-wise component contributions to the output.** The y-axis value ($\alpha$) represents the weight assigned to the Mamba component, while $1 - \alpha$ denotes the coefficient for the global attention module. We measure these values using a 12-layer 350M intra-hybrid model.

To provide a deeper analysis, we visualize the token-wise contributions in Table 12. We examine random samples to identify which module is strongly activated at specific layers for next-token prediction. There are several common and intriguing patterns. First, while the Mamba component plays a dominant role overall in the first layer, the Transformer component tends to be utilized more heavily at sentence boundaries—marked by punctuation such as periods (.) or question marks (?)—or at positions requiring the start of new information.

Layer 6 exhibits a particularly interesting pattern: the model employs global attention to predict the next token when retrieving key context keywords or predicting numerical values and complex information. For example, in the first sample, global attention is activated to derive updated temporal information based on the context following the mention of a specific date like 'Oct 7'12'.

In the subsequent Layer 7, the Transformer component is frequently activated at the start of new syntactic units, such as after conjunctions, periods, or parentheses. Conversely, when a single word is split into sub-words by the tokenizer, the SSM component activates, leveraging its ability to process local information more efficiently.

Table 12: **Visualization of token-wise contribution from Transformer and Mamba modules.** We use an intra-layer hybrid 350M model where every layer consists of an intra-hybrid block. The fusion mechanism combines the outputs of each module by weighting them with $\alpha$ and $1-\alpha$, respectively. Red color corresponds to a larger coefficient for the Transformer component, and Blue corresponds to a larger coefficient for the Mamba component. For a model consisting of 12 layers in total, we visualize for three random samples at the first layer (index 0) and the middle layers (indices 6 and 7). Note that these coefficients are specifically optimized for *next*-token prediction.

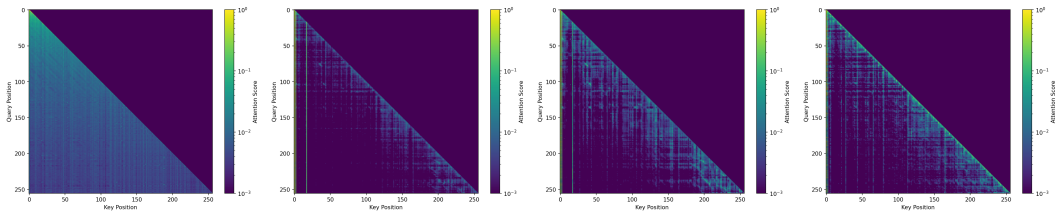
| Layer | Text |
|---|---|
| Layer #0 | <\|begin_of_text\|> Take the 2 -minute tour × Here what happened with me today . Time Machine asked me whether I want to set a backup disk , I've answered yes , but then , when I've realized that in order to backup anything Time Machine will clean the disk , I've changed my mind and canceled everything . And my disk suddenly became read only . What I've tried before Go og ling : $ sudo ch flags - R n ouch g Elements / $ sudo chmod - R a +w Elements / But I've failed with both of this , getting " read -only file system " messages . What I've tried after Go og ling : 1 . Open Disk Utilities 2 . Click Repair Disk Permissions But this button is disabled , and I have no idea what exactly should be done to enable it . I have been using this disk for a quite a long time , and never had any permission issues with it . ( Disk is formatted as NT FS , if that helps . Capacity is 2 TB , of which 1 . 92 TB are available .) I'd really appreciate if someone will give me a hint how this can be resolved . share \| im prove this question Try boot ing from the Recovery HD and see if the button is still disabled . – rien 333 Oct 7 ' 12 at 14 : 07 What exactly is it you need to do with the stuff on the NT FS disk ? – seg idd ins Oct 7 ' 12 at 15 : 26 @ Sam uel E .G idd ins I need to work with disk just as I've worked yesterday , 2 days ago , 3 days ago etc . It looks like very annoying bug to me - to find out that out of a sudden this disk is considered to be read -only . – sh ab unc Oct 7 ' 12 at 15 : 34 do you have any add -ons that would |
| Layer #6 | <\|begin_of_text\|> Take the 2 -minute tour × Here what happened with me today . Time Machine asked me whether I want to set a backup disk , I've answered yes , but then , when I've realized that in order to backup anything Time Machine will clean the disk , I've changed my mind and canceled everything . And my disk suddenly became read only . What I've tried before Go og ling : $ sudo ch flags - R n ouch g Elements / $ sudo chmod - R a +w Elements / But I've failed with both of this , getting " read -only file system " messages . What I've tried after Go og ling : 1 . Open Disk Utilities 2 . Click Repair Disk Permissions But this button is disabled , and I have no idea what exactly should be done to enable it . I have been using this disk for a quite a long time , and never had any permission issues with it . ( Disk is formatted as NT FS , if that helps . Capacity is 2 TB , of which 1 . 92 TB are available .) I'd really appreciate if someone will give me a hint how this can be resolved . share \| im prove this question Try boot ing from the Recovery HD and see if the button is still disabled . – rien 333 Oct 7 ' 12 at 14 : 07 What exactly is it you need to do with the stuff on the NT FS disk ? – seg idd ins Oct 7 ' 12 at 15 : 26 @ Sam uel E .G idd ins I need to work with disk just as I've worked yesterday , 2 days ago , 3 days ago etc . It looks like very annoying bug to me - to find out that out of a sudden this disk is considered to be read -only . – sh ab unc Oct 7 ' 12 at 15 : 34 do you have any add -ons that would |
| Layer #7 | <\|begin_of_text\|> Take the 2 -minute tour × Here what happened with me today . Time Machine asked me whether I want to set a backup disk , I've answered yes , but then , when I've realized that in order to backup anything Time Machine will clean the disk , I've changed my mind and canceled everything . And my disk suddenly became read only . What I've tried before Go og ling : $ sudo ch flags - R n ouch g Elements / $ sudo chmod - R a +w Elements / But I've failed with both of this , getting " read -only file system " messages . What I've tried after Go og ling : 1 . Open Disk Utilities 2 . Click Repair Disk Permissions But this button is disabled , and I have no idea what exactly should be done to enable it . I have been using this disk for a quite a long time , and never had any permission issues with it . ( Disk is formatted as NT FS , if that helps . Capacity is 2 TB , of which 1 . 92 TB are available .) I'd really appreciate if someone will give me a hint how this can be resolved . share \| im prove this question Try boot ing from the Recovery HD and see if the button is still disabled . – rien 333 Oct 7 ' 12 at 14 : 07 What exactly is it you need to do with the stuff on the NT FS disk ? – seg idd ins Oct 7 ' 12 at 15 : 26 @ Sam uel E .G idd ins I need to work with disk just as I've worked yesterday , 2 days ago , 3 days ago etc . It looks like very annoying bug to me - to find out that out of a sudden this disk is considered to be read -only . – sh ab unc Oct 7 ' 12 at 15 : 34 do you have any add -ons that would |

| Layer | Text |
|---|---|
| Layer #0 | Hol zer at j ess ica .h ol zer @d ow j ones .com <\|end_of_text\|> <\|begin_of_text\|> When the Associated Press releases its 200 9 preseason college football poll this weekend , it 's all but guaranteed that Florida will be ranked No . 1 —and that the G ators will collect a fat majority of the first -place votes . Florida already received 90 % of first -place votes in the recent coaches ' poll . Florida quarterback Tim Te bow Associated Press It 's no mystery why . The G ators are the defending B CS national champions and their quarterback , Tim Te bow , and their entire starting defense is coming back . Rare ly has a defending champion enjoyed such riches . But there 's some evidence that Florida 's fans should think twice before ordering their commem orative " Repeat Champions !" T -shirts . Bad O men for the G ators Here are the 10 teams that received the highest percentages of first -place votes in the preseason AP football poll since 196 8 —and how they finished the season . 1 . USC , 200 7 ( 95 . 4 %) 11 - 2 ( 3 rd ) 6 . Florida State , 199 1 ( 81 . 7 %) 11 - 2 ( 4 th ) 2 . USC , 200 5 ( 92 . 3 %) 12 - 1 ( 2 nd ) 7 . Ohio State , 196 9 ( 78 . 8 %) 8 - 1 ( 4 th ) 3 . Oklahoma , 198 7 ( 91 . 7 %) 11 - 1 ( 3 rd ) 8 . Nebraska , 199 6 ( 74 . 6 %) 11 - 2 ( 6 th ) 4 . Oklahoma , 197 5 ( 90 %) 11 - 1 ( 1 st ) 9 . USC , 197 9 ( 74 . 6 %) 11 - 0 - 1 ( 2 nd ) 5 . USC , 197 3 ( 87 . 3 %) 9 - 2 - 1 ( 8 th ) 10 . Oklahoma , 198 6 ( 74 . 6 %) 11 - 1 ( 3 rd ) Since 196 8 , 10 teams have received at least 75 % of the first -place votes in the preseason AP poll . Like Florida , seven of them were defending national champions . But only one of those teams , Oklahoma in 197 5 , actually won the national championship . None of these teams went undefeated —in fact , their average |
| Layer #6 | Hol zer at j ess ica .h ol zer @d ow j ones .com <\|end_of_text\|> <\|begin_of_text\|> When the Associated Press releases its 200 9 preseason college football poll this weekend , it 's all but guaranteed that Florida will be ranked No . 1 —and that the G ators will collect a fat majority of the first -place votes . Florida already received 90 % of first -place votes in the recent coaches ' poll . Florida quarterback Tim Te bow Associated Press It 's no mystery why . The G ators are the defending B CS national champions and their quarterback , Tim Te bow , and their entire starting defense is coming back . Rare ly has a defending champion enjoyed such riches . But there 's some evidence that Florida 's fans should think twice before ordering their commem orative " Repeat Champions !" T -shirts . Bad O men for the G ators Here are the 10 teams that received the highest percentages of first -place votes in the preseason AP football poll since 196 8 —and how they finished the season . 1 . USC , 200 7 ( 95 . 4 %) 11 - 2 ( 3 rd ) 6 . Florida State , 199 1 ( 81 . 7 %) 11 - 2 ( 4 th ) 2 . USC , 200 5 ( 92 . 3 %) 12 - 1 ( 2 nd ) 7 . Ohio State , 196 9 ( 78 . 8 %) 8 - 1 ( 4 th ) 3 . Oklahoma , 198 7 ( 91 . 7 %) 11 - 1 ( 3 rd ) 8 . Nebraska , 199 6 ( 74 . 6 %) 11 - 2 ( 6 th ) 4 . Oklahoma , 197 5 ( 90 %) 11 - 1 ( 1 st ) 9 . USC , 197 9 ( 74 . 6 %) 11 - 0 - 1 ( 2 nd ) 5 . USC , 197 3 ( 87 . 3 %) 9 - 2 - 1 ( 8 th ) 10 . Oklahoma , 198 6 ( 74 . 6 %) 11 - 1 ( 3 rd ) Since 196 8 , 10 teams have received at least 75 % of the first -place votes in the preseason AP poll . Like Florida , seven of them were defending national champions . But only one of those teams , Oklahoma in 197 5 , actually won the national championship . None of these teams went undefeated —in fact , their average |
| Layer #7 | Hol zer at j ess ica .h ol zer @d ow j ones .com <\|end_of_text\|> <\|begin_of_text\|> When the Associated Press releases its 200 9 preseason college football poll this weekend , it 's all but guaranteed that Florida will be ranked No . 1 —and that the G ators will collect a fat majority of the first -place votes . Florida already received 90 % of first -place votes in the recent coaches ' poll . Florida quarterback Tim Te bow Associated Press It 's no mystery why . The G ators are the defending B CS national champions and their quarterback , Tim Te bow , and their entire starting defense is coming back . Rare ly has a defending champion enjoyed such riches . But there 's some evidence that Florida 's fans should think twice before ordering their commem orative " Repeat Champions !" T -shirts . Bad O men for the G ators Here are the 10 teams that received the highest percentages of first -place votes in the preseason AP football poll since 196 8 —and how they finished the season . 1 . USC , 200 7 ( 95 . 4 %) 11 - 2 ( 3 rd ) 6 . Florida State , 199 1 ( 81 . 7 %) 11 - 2 ( 4 th ) 2 . USC , 200 5 ( 92 . 3 %) 12 - 1 ( 2 nd ) 7 . Ohio State , 196 9 ( 78 . 8 %) 8 - 1 ( 4 th ) 3 . Oklahoma , 198 7 ( 91 . 7 %) 11 - 1 ( 3 rd ) 8 . Nebraska , 199 6 ( 74 . 6 %) 11 - 2 ( 6 th ) 4 . Oklahoma , 197 5 ( 90 %) 11 - 1 ( 1 st ) 9 . USC , 197 9 ( 74 . 6 %) 11 - 0 - 1 ( 2 nd ) 5 . USC , 197 3 ( 87 . 3 %) 9 - 2 - 1 ( 8 th ) 10 . Oklahoma , 198 6 ( 74 . 6 %) 11 - 1 ( 3 rd ) Since 196 8 , 10 teams have received at least 75 % of the first -place votes in the preseason AP poll . Like Florida , seven of them were defending national champions . But only one of those teams , Oklahoma in 197 5 , actually won the national championship . None of these teams went undefeated —in fact , their average |

| Layer | Text |
|---|---|
| Layer #0 | team lacks a Rapid Spinner . And some Pokemon who fail to 2 H KO my Pokemon will 2 H KO them with enough hazards on my team which could lead to me being swept . It would also take care of my Ag gr on weakness . The problem is , I don 't see anything I can replace it with ... if someone could give me a good enough reason on why I should replace it with " X " Pokemon or why I shouldn 't , that would be great . · Threat List ( Sorry E o but I j acked this from your R MT ; ___ ; ) Red means this Pokemon is a big threat . Blue means this Pokemon is a moderate threat . Black means this Pokemon is easily handled . U U Threat s Abs ol - Mil otic and We ez ing can take a + 2 Attack , I just have to hope it doesn 't crit either of them , and hopefully I have entry hazards up so it dies quickly from LO . Ag gr on - This Pokemon is a MA JOR threat . I can 't switch in anything on it , as it 2 H KO es everything . If it comes in on Chan sey / Cle f able , I have to sacrifice them as I cant switch in Mil otic and risk Ag gr on being J olly and being 2 H KO d . A smart player can keep switching it in and out until my special walls are gone . Al ak az am - Switch to Spirit omb , Purs uit it and it 's KO ed . Not a big threat at all . Chan sey can take it on as well , even though a Specs Focus Blast is going to hurt . ( No ob th und and your Zam nom : P < 3 ) Alt aria - Cle f able can Encore / Trick DD variants , Mil otic can H aze / Ice Beam DD variants as well . Support variants are handled by Cle f able easily ; just Encore and Se ismic T oss it until it 's KO ed . · Az umar ill - Choice Band variants are easily handled by Mil otic , We ez ing , and Spirit omb . Sub P unch variants can be Enc ored by Cle f able and taken on by We ez ing . · Bl az iken - Mil otic is my only |
| Layer #6 | team lacks a Rapid Spinner . And some Pokemon who fail to 2 H KO my Pokemon will 2 H KO them with enough hazards on my team which could lead to me being swept . It would also take care of my Ag gr on weakness . The problem is , I don 't see anything I can replace it with ... if someone could give me a good enough reason on why I should replace it with " X " Pokemon or why I shouldn 't , that would be great . · Threat List ( Sorry E o but I j acked this from your R MT ; ___ ; ) Red means this Pokemon is a big threat . Blue means this Pokemon is a moderate threat . Black means this Pokemon is easily handled . U U Threat s Abs ol - Mil otic and We ez ing can take a + 2 Attack , I just have to hope it doesn 't crit either of them , and hopefully I have entry hazards up so it dies quickly from LO . Ag gr on - This Pokemon is a MA JOR threat . I can 't switch in anything on it , as it 2 H KO es everything . If it comes in on Chan sey / Cle f able , I have to sacrifice them as I cant switch in Mil otic and risk Ag gr on being J olly and being 2 H KO d . A smart player can keep switching it in and out until my special walls are gone . Al ak az am - Switch to Spirit omb , Purs uit it and it 's KO ed . Not a big threat at all . Chan sey can take it on as well , even though a Specs Focus Blast is going to hurt . ( No ob th und and your Zam nom : P < 3 ) Alt aria - Cle f able can Encore / Trick DD variants , Mil otic can H aze / Ice Beam DD variants as well . Support variants are handled by Cle f able easily ; just Encore and Se ismic T oss it until it 's KO ed . · Az umar ill - Choice Band variants are easily handled by Mil otic , We ez ing , and Spirit omb . Sub P unch variants can be Enc ored by Cle f able and taken on by We ez ing . · Bl az iken - Mil otic is my only |
| Layer #7 | team lacks a Rapid Spinner . And some Pokemon who fail to 2 H KO my Pokemon will 2 H KO them with enough hazards on my team which could lead to me being swept . It would also take care of my Ag gr on weakness . The problem is , I don 't see anything I can replace it with ... if someone could give me a good enough reason on why I should replace it with " X " Pokemon or why I shouldn 't , that would be great . · Threat List ( Sorry E o but I j acked this from your R MT ; ___ ; ) Red means this Pokemon is a big threat . Blue means this Pokemon is a moderate threat . Black means this Pokemon is easily handled . U U Threat s Abs ol - Mil otic and We ez ing can take a + 2 Attack , I just have to hope it doesn 't crit either of them , and hopefully I have entry hazards up so it dies quickly from LO . Ag gr on - This Pokemon is a MA JOR threat . I can 't switch in anything on it , as it 2 H KO es everything . If it comes in on Chan sey / Cle f able , I have to sacrifice them as I cant switch in Mil otic and risk Ag gr on being J olly and being 2 H KO d . A smart player can keep switching it in and out until my special walls are gone . Al ak az am - Switch to Spirit omb , Purs uit it and it 's KO ed . Not a big threat at all . Chan sey can take it on as well , even though a Specs Focus Blast is going to hurt . ( No ob th und and your Zam nom : P < 3 ) Alt aria - Cle f able can Encore / Trick DD variants , Mil otic can H aze / Ice Beam DD variants as well . Support variants are handled by Cle f able easily ; just Encore and Se ismic T oss it until it 's KO ed . · Az umar ill - Choice Band variants are easily handled by Mil otic , We ez ing , and Spirit omb . Sub P unch variants can be Enc ored by Cle f able and taken on by We ez ing . · Bl az iken - Mil otic is my only |

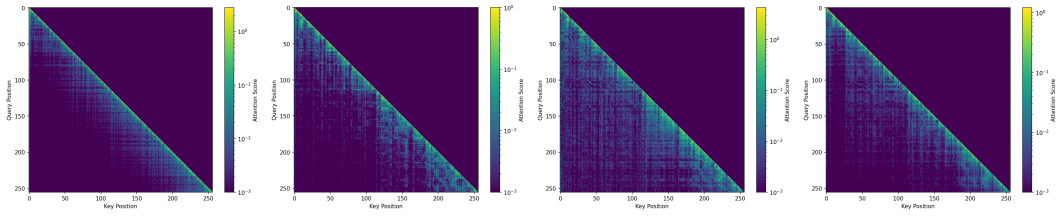
## J VISUALIZATION OF ATTENTION SCORES

Figures 7 and 8 visualize the attention scores of Transformer and Mamba, respectively. For Mamba, we visualize the implicit attention map of the SSM module by following the approach of Deci-Mamba (Ben-Kish et al., 2024). In the case of the Transformer, the attention scores in the initial layers show a highly uniform distribution, suggesting that these layers primarily serve to encode the hidden states. As a result, using Transformer blocks in the early stages of an inter-layer hybrid architecture may negatively impact performance due to this uniform attending behavior. In contrast, Mamba exhibits a strong focus along the diagonal, resulting in a very small effective receptive fields (ERFs). This is the opposite of the Transformer's tendency to attend to a wider range of token positions. Consequently, hybridization appears to improve quality by providing an ensemble-like effect.



(a) Layer 1 (b) Layer 6 (c) Layer 11 (d) Layer 16

Figure 7: **Qualitative results of attention scores from Llama 1B model.** We measure the score at a context length of 256 using the PG19 test set. The evaluation is conducted with a Llama 1B model trained on 60B tokens. Values closer to yellow indicate scores near 1, while values closer to purple indicate scores near 0.



(a) Layer 1 (b) Layer 5 (c) Layer 9 (d) Layer 13

Figure 8: **Qualitative results of attention scores from Mamba 1B model.** We measure the score at a context length of 256 using the PG19 test set. The evaluation is conducted with a Mamba 1B model trained on 60B tokens. Values closer to yellow indicate scores near 1, while values closer to purple indicate scores near 0. We take the absolute value of the implicit attention scores in Mamba and normalize each row by its maximum value.

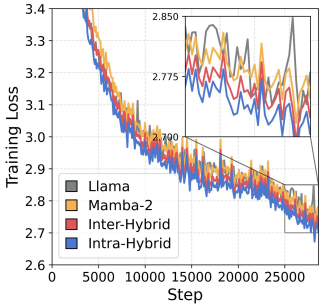
## K Training Dynamics for Hybrid Architectures

The left panel of Figure 13 illustrates the learning curves used to analyze convergence speed and stability. Initially, Mamba converges more slowly than the Transformer; however, it eventually catches up to Llama in terms of training loss over the course of 60B tokens. Conversely, the Hybrid models match Llama's loss trajectory initially but begin to outperform it (showing lower loss) from the early-to-mid stages. We hypothesize that the combination of these primitives provides the model with greater optimization potential. Furthermore, regarding training stability, we observe no distinct deviations across the architectures; all models exhibit similar fluctuation patterns throughout the training process (only Llama shows slightly different waveform because it was trained using 16 GPUs (DP = 16), whereas the other models were trained on 8 GPUs.

## L Finetuning Performance on Downstream Tasks

Extending beyond language modeling and retrieval, we evaluate the five 1B checkpoints (Table 2)—all pre-trained with an identical 4e20 FLOPs budget—on downstream summarization (Sum) and question-answering (QA). The benchmarks include Multi-News (MN; Fabbri et al. (2019)), BIGPATENT (BP; Sharma et al. (2019)), and GovReport (GR; Huang et al. (2021)) for summarization, alongside SQuAD (SQA; Rajpurkar et al. (2016)), NarrativeQA (NQA; Kočiský et al. (2018)), and TriviaQA (TQA; Joshi et al. (2017)) for QA. We fine-tune for 1,000 steps with a global batch size of 16 using a trapezoidal scheduler (peak learning rate of 1e-4). The sequence length is set to 8K for all datasets, except SQuAD (2K). We allocate maximum target (summaries or question-answer pairs) lengths of 512 (MN, BP), 768 (GR), and 128 (all QA datasets), dedicating the remaining capacity to the input context. Unused slots are padded with `EOS` tokens and masked out from the attention mechanism. Evaluation is performed on 1,000 sampled test instances for each dataset, with the exception of SQuAD, where the validation set is used.

The right panel of Table 13 summarizes the performance on downstream tasks. Although SWA and Mamba appeared competitive with the standard Transformer in language modeling, this competence does not fully translate to downstream tasks. Specifically, in summarization tasks requiring contextual retrieval—mirroring our NIAH findings—SWA and Mamba suffer significant performance degradation due to their strong locality biases. While the shorter generation nature of QA tasks allows these models to rival the Transformer on some datasets, they still face substantial performance drops on specific benchmarks. In contrast, hybrid architectures demonstrate remarkable robustness regardless of the integration strategy (inter- or intra-hybrid), consistently matching or outperforming the vanilla Transformer. This confirms that hybrid models effectively achieve superior model quality with greater efficiency, overcoming the limitations of homogeneous Mamba models.




| | Base Config | | | | | Sum Tasks | | | QA Tasks | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Models** | N-emb | $N_{\text{attn}}$ | $N_{\text{swa}}$ | $N_{\text{ssm}}$ | $N_{\text{hyb}}$ | MN | BP | GR | SQA | NQA | TQA |
| Llama | 0.97B | 16 | - | - | - | 18.66 | 32.15 | 20.55 | 54.72 | 17.35 | 51.89 |
| SWA | 0.97B | 3 | 13 | - | - | 15.87 | 29.74 | 18.91 | 48.55 | 17.52 | **51.99** |
| Mamba | 0.99B | - | - | 13 | - | 16.40 | 12.66 | 13.08 | **55.55** | 16.64 | 36.50 |
| Inter-H | 0.96B | 2 | - | 11 | - | **19.00** | **32.76** | 19.56 | 54.91 | 17.39 | 51.66 |
| Intra-H | 0.98B | - | - | 11 | 2 | 18.33 | 31.80 | **20.76** | 55.42 | **18.48** | 51.00 |

Table 13: ***(Left)*** **Learning curves across four different architectures.** The models are pretrained on 60B tokens using the DCLM dataset, with checkpoints logged every 100 steps. For the hybrid model, we adopt a block ratio of 1:5. ***(Right)*** **Comparison of fine-tuning performance across downstream summarization and question-answering (QA) tasks.** We fine-tune five distinct architectures, all pre-trained with a compute budget of 4.5e20 FLOPs. Our evaluation benchmarks include Multi-News, BIGPATENT, and GovReport for summarization, alongside SQuAD, NarrativeQA, and TriviaQA for QA. For hybrid architectures that integrate sliding window attention, Mamba, or intra-hybrid blocks with Transformer blocks, we maintain a consistent block mixing ratio of 1:5.