Physics of Language Models: Part 4.1, Architecture Design and the Magic of Canon Layers

[extended abstract]*

Zeyuan Allen-Zhu
FAIR at Meta
physics.allen-zhu.com

Abstract

Understanding architectural differences in language models is challenging, especially at academic-scale pretraining (e.g., 1.3B parameters, 100B tokens), where results are often dominated by noise and randomness. To overcome this, we introduce controlled synthetic pretraining tasks that isolate and evaluate core model capabilities. Within this framework, we discover *Canon layers*: lightweight architectural components—named after the musical term "canon"—that promote horizontal information flow across neighboring tokens. Canon layers compute weighted sums of nearby token representations and integrate seamlessly into Transformers, linear attention, state-space models, or any sequence architecture.

We present 12 key results. This includes how Canon layers enhance reasoning depth (e.g., by $2\times$), reasoning breadth, knowledge manipulation, etc. They lift weak architectures like NoPE to match RoPE, and linear attention to rival SOTA linear models like Mamba2/GDN—validated both through synthetic tasks and real-world academic-scale pretraining. This synthetic playground offers an *economical, principled path* to isolate core model capabilities often obscured at academic scales. Equipped with infinite high-quality data, it may even *predict* how future architectures will behave as training pipelines improve—e.g., through better data curation or RL-based post-training—unlocking deeper reasoning and hierarchical inference.

1 Introduction

Recent advances in large language models (LLMs) have sparked transformative progress across numerous tasks, including question answering, summarization, translation, code generation [13, 15, 39, 61]. Despite rapid progress, systematic understanding of effective neural architecture design has remained elusive, fundamentally hindered by some major challenges.

Challenge 1: Pretraining loss as an unreliable proxy for intelligence. Architectural comparisons often rely on perplexity or cross-entropy loss, but these metrics do not reliably reflect real-world capabilities—especially since natural data is *skills-mixed*. For example, state-space architectures like Mamba [19, 26] frequently achieve lower perplexity early in training due to rapid memorization, yet perform poorly on complex reasoning tasks. Reliance on *early stopping via perplexity* is thus problematic: it may lead to comparing models that have merely internalized surface-level linguistic

*__Following the theory community tradition, we defer the full and future editions of this paper__ to our project page physics.allen-zhu.com and ssrn.com/abstract=5240330. The full V1.1 paper underwent NeurIPS review; however, due to the density of results, we recommend consulting the full version for readability. Synthetic GatedDeltaNet (GDN) experiments were newly added in V2.0. Results for 1–8B Canon-layer-pretrained models on real-world data are open-sourced on our website (physics.allen-zhu.com). These were not included in our original NeurIPS 2025 submission, and we reserve the right to submit them elsewhere.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

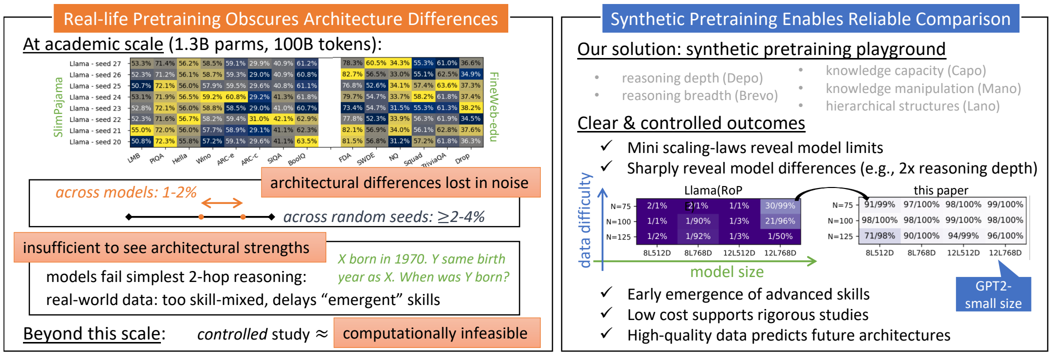

Figure 1: Architecture search in noisy real-life pretrain (good luck!) vs. synthetic playground (scientific rigor).

patterns without developing deeper reasoning or factual understanding [31].

Challenge 2: Noise below emergence thresholds. Emergent abilities—complex skills that only arise in large-scale models (e.g., 7B parameters, 10T tokens [1])—complicate architectural comparisons at smaller, academic scales (e.g., 1.3B parameters, 100B tokens [9, 25, 70]). At these scales, small benchmark gains (e.g., 2%) often result from random initialization or data shuffling—variance that can cause 2–4% swings in accuracy (see Figure 1). More fundamentally, **models fail even the simplest 2-hop reasoning** tasks, performing no better than random guessing.[2] This basic reasoning floor *masks architectural differences* in more advanced cognitive skills, making evaluation at this scale deeply unreliable. While large-scale industry training might reveal these differences, its prohibitive cost blocks systematic ablations, impeding academic contributions to rigorous architecture science—and often reducing design choices to heuristics and guesswork.

Challenge 3: Grokking, Data Quality, and Curriculum Learning. Failures in complex reasoning tasks typically stem from deficiencies in training data, *not* architectural limitations. Too few challenging samples and a lack of intermediate-complexity data often force models to rely on unstable grokking behavior—where generalization only emerges after unnecessarily long pretraining [43]—and disrupt curriculum learning [10]. For instance, models lacking 2-hop reasoning data may unpredictably learn 3-hop tasks after extensive exposure to 1-hop and 3-hop examples. This makes training highly sensitive to randomness, further complicating architectural comparisons. Reinforcement learning (RL)-based post-training methods, such as GRPO [53] and PPO [52], aim to address this by delivering tailored data at optimal difficulty levels. While effective, these methods introduce new experimental confounds—it becomes unclear whether performance gains stem from pretraining, RL fine-tuning, stochastic training dynamics, or architectural strength.

Our approach: Atomic decomposition of intelligence. To overcome the noise and cost of real-world pretraining—especially at academic scales where even 2-hop reasoning fails to emerge—we decompose intelligence into core (ideally atomic!) components, such as reasoning depth and breadth, and design synthetic, controllable *pretrain* tasks to isolate and evaluate them independently. This framework sharply characterizes architectural strengths and scalability under clean, idealized conditions (see Figure 1), offering a principled and economical path for architecture design.

This approach directly addresses Challenge 1 by enabling *single-skill evaluations*, minimizing the confounding factors prevalent in real-world pretraining data. For example, it allows rigorous comparisons of whether architecture A outperforms architecture B in reasoning depth, while ensuring modifications do not degrade other capabilities. By isolating intrinsic architectural biases, synthetic *pretrain* tasks reveal properties often obscured by noise and mixed signals in typical real-life setups.

Challenge 2 is mitigated by *lowering resource* needs for rigorous comparisons. Synthetic benchmarks yield infinite high-quality data, enabling meaningful pretraining even for smaller models (e.g., GPT2-small) where complex skills might otherwise not emerge. In these controlled environments, capabilities like deep multi-hop reasoning *emerge clearly and reliably*, allowing rapid identification of architectural limitations, investigation of *mini scaling-laws*, and uncover trends that real-world pretrained models often fail to reveal due to noise or insufficient signal despite extensive training.

For Challenge 3, we manage data difficulty distributions to ensure adequate representation of intermediate-complexity samples, smoothing learning curves and enabling the *early and consis-*

[2]In our simplest 2-hop reasoning tasks, birth years for 3 individuals are presented, followed by 3 "[name2] was born in the same year as [name1]" equivalences. The model is prompted to infer the second group's birth years. Academic-scale models can only guess. See Result 12.

tent emergence of advanced skills—unlike less predictable real-world data prone to grokking-driven instability. As training pipelines improve—via better data curation or RL-based continued pretraining—synthetic pretrain benchmarks may provide *predictive insight* into which architectures best support scaling to more advanced tasks in the future.

We draw inspiration from physics, where idealized settings—such as frictionless planes or vacuum chambers—reveal first principles by removing confounding factors. Similarly, synthetic tasks eliminate the noise, randomness, and data contamination of real-world datasets, enabling clean, controlled, apples-to-apples architectural comparisons, much like Galileo's Pisa tower experiment.

This paper's key contributions are summarized below:

Result 0: Building the Synthetic Playground (Section 2+3). We introduce five synthetic pretraining tasks—DEPO (reasoning depth), BREVO (reasoning breadth), CAPO (knowledge capacity), MANO (knowledge manipulation), and LANO (hierarchical language structure). This controlled environment can reveal clear, commonsense capability trends *at smaller scales*: linear attention (e.g., GLA [69]) consistently underperforms; state-space models like Mamba2 [19] excel at memory but struggle with reasoning; and full Transformers dominate on complex reasoning tasks.

Result 1: Canon Layers Add Horizontal Information Flow (see full paper) . Transformers lack horizontal information flow within layers, leading to inefficiencies even on simple tasks like associative recall. Drawing on the musical canon (overlapping repetition), we introduce *Canon layers*, horizontal "residual links" across neighboring tokens that can be flexibly inserted at multiple points — before attention (Canon-A), inside attention (Canon-B), before MLP (Canon-C), inside MLP (Canon-D). While Canon layers can be implemented in many ways—even simple random averaging is highly effective—this paper focuses on trainable 1-d linear convolutions of kernel size 4. This is lightweight and integrates seamlessly into any sequence model with minimal code.

Results 2–5: When Transformer Meets Canon (see full paper) .

- BOOST PERFORMANCE. In our playground, Canon layers improve reasoning depth (200–400%), reasoning breadth (30%), knowledge manipulation length (30%), and more. These stem from enhanced hierarchical learning dynamics and come with minimal computational overhead.
- REVIVING NOPE. Integrating Canon layers transforms NoPE models into strong performers, often matching or surpassing RoPE(+Canon). Canon layers outperform positional fixes like ALiBi [44] or H-Alibi [30], and reducing/removing RoPE usage improves length generalization.
- ABLATION STUDY. Canon layers contribute cumulatively across sublayer positions (Canon-A/B/C/D), independently of attention or MLP components. Residual links improve training efficiency; minimal parameter tuning is required without compromising stability.
- MLP AND MOE. Canon layers can recover some knowledge capacity lost in gated MLP or mixture-of-expert (MoE) architectures, via improved training efficiency and stability.

Results 6–7: When Linear Attention Meets Canon (see full paper) .

- BOOST PERFORMANCE. Canon layers elevate Gated Linear Attention (GLA [69]) from 1-hop to 4-hop reasoning depth, double its reasoning breadth and knowledge manipulation length, making it comparable to Mamba2 and even surpassing it on tasks like BREVO.
- ABLATION STUDY. Residual links and full Canon (A/B/C/D) are essential for maximizing effectiveness for linear-attention models, partial implementations may underperform.

Results 8–9: When Mamba Meets Canon (see full paper) .

- SECRET OF SUCCESS. Mamba2's performance is driven by its built-in `conv1d` mechanism, which acts as a non-linear Canon-B layer applied to selective coordinates. Removing `conv1d` drops performance to match GLA, while replacing it with full Canon layers further boosts results, highlighting the importance of horizontal information flow over SSM design.
- ABLATION STUDY. Canon choices—such as integration points and residual links—can influence Mamba2's performance. Mimetic initialization [63], while optimized for length generalization, harms shorter-context tasks, underscoring the need for diverse pretraining environments.

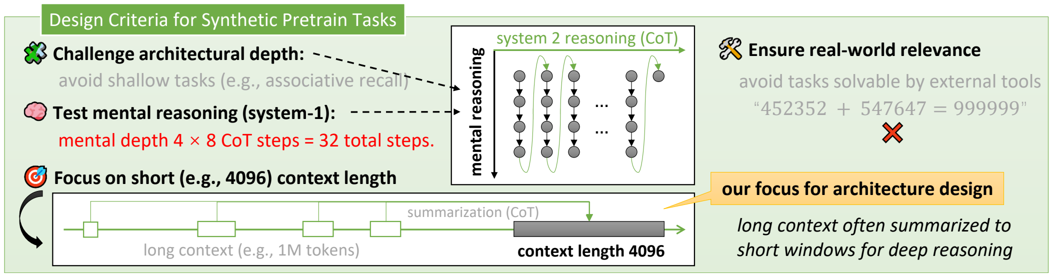

Figure 2: Our design criteria for synthetic pretrain tasks.

Results 10–11: Comparing Architectures (see full paper) .

- CONTROLLED COMPARISONS. Applying full Canon layers consistently across RoPE, NoPE, Mamba2, and GLA allows controlled comparisons, revealing that full transformers outperform linear models in hierarchical reasoning tasks, achieving twice the reasoning depth.
- REASONING DEPTH CHALLENGES. In GLA and Mamba2, limited reasoning depth stems from accumulated compression and retrieval errors—not memory capacity—pinpointing a key focus for future research on linear models. Until this is resolved, hybrid designs (e.g., sliding-window Transformers with linear backbones) remain the most scalable path to deeper reasoning.

Result 12: Academic-Scale Real-World Pretraining (see full paper) . Training 1.3B-parameter models on 100B tokens (context length 4096) reveals high noise and limited resolution, making many architectural comparisons statistically unreliable. Still, several consistent patterns emerge. Canon layers significantly improve NoPE and GLA—elevating them to match RoPE and Mamba2, respectively—while removing `conv1d` weakens Mamba2 to GLA level. Linear models lag behind full Transformers on retrieval-heavy tasks, even with Canon layers. All models fail 2-hop reasoning, even in short contexts (e.g., 100 tokens), underscoring the limitations of academic-scale pretraining. Reducing or removing RoPE improves long-context generalization when Canon layers are present. These results align with our synthetic findings (Results 3, 6, 8, 10, 11).

In summary, Canon layers fundamentally improve horizontal information flow across diverse architectures, enabling deeper reasoning and efficient scalability. Combined with synthetic benchmarks, they provide systematic insights into future opportunities in model design.

Future research. We plan to explore applications of Canon layers beyond academic scale, whose preliminary findings (w.r.t. 1-8B models pretrained using 1-2T tokens) align closely with those in this paper. Code is available on GitHub, models on HuggingFace, and all links are provided at `physics.allen-zhu.com`.

2 Synthetic Tasks for Decomposing Intelligence

We design synthetic tasks to systematically evaluate specific capabilities of language model architectures under controlled conditions, minimizing confounds and enabling clean comparisons. Task selection is guided by four criteria:

Criterion 1: Tasks must not be shallow. Shallow tasks—like associative recall or copying—are easily solvable by small and shallow models, and do not meaningfully test architectural strength. Deep learning relies on stacked layers to progressively learn abstract features [4], so tasks involving hierarchical reasoning better evaluate architectural scalability and efficiency.

Criterion 2: Emphasis on mental thinking. Tasks should assess a model's ability to reason internally without Chain-of-Thought (CoT). While CoT helps decompose problems, it does not reflect intrinsic "system 1" reasoning [74]. For example, a model reasoning 4 steps internally and 8 via CoT achieves 32 steps, but *only internal ones reflect architectural strength*. Current models like o3/R1 produce verbose reasoning traces even for trivial prompts (e.g., "Hello")—revealing inefficiencies in system 1. To guide architectural progress, tasks must target mental reasoning.

Criterion 3: Avoid emphasis on length generalization. Length generalization is often unstable—sensitive to random seeds and training order [79]—and thus unreliable for comparing architectures. While length generalization is important, models over-optimized for long contexts (e.g., 100k

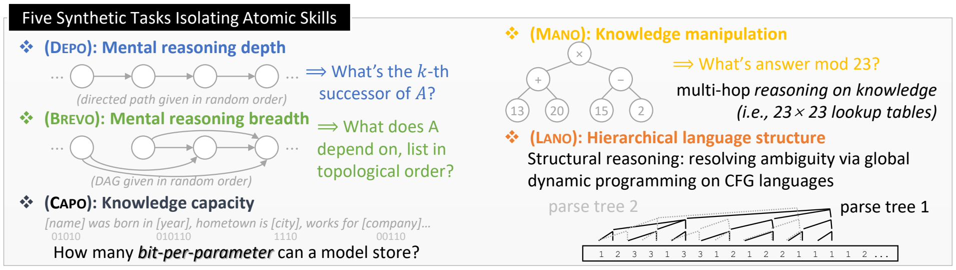

Figure 3: Overview of our five synthetic tasks, each isolating an atomic skill for rigorous architectural comparison.

tokens) may exhibit reduced performance on standard lengths like 4096 tokens.[3] In practice, long inputs are typically summarized into shorter windows before reasoning, so we prioritize evaluating architectures on dense, 4096-token contexts, where critical reasoning unfolds.

Criterion 4: Relevance to real-world skills. Tasks should prioritize broadly applicable skills while avoiding capabilities better suited to external tools. For example, large-number arithmetic (e.g., adding 10-digit numbers) is theoretically interesting but can be delegated to Python interpreters; failures in this area typically reflect limited data exposure rather than architectural weaknesses (e.g., Llama3 70B miscalculates 452352 + 547647). Synthetic tasks should focus on universally relevant skills, aligned with real-world applications, to ensure meaningful assessments.

2.1 Our First Set of Five Synthetic Pretrain Tasks

To operationalize the criteria above, we design five synthetic tasks—each targeting a distinct dimension of language model capability. We name them DEPO, BREO, CAPO, MANO, and LANO.

Task DEPO: Mental reasoning depth. Reasoning depth represents a fundamental capability for LLMs, requiring models to retrieve information through multi-step computation. Task DEPO evaluates reasoning depth as k-hop traversal over directed permutations, where models compute the k-th successor for each query q entirely internally, without intermediate steps like Chain-of-Thought (CoT).[4] Each instance is formatted as:

```
<bos> x1 y1 x2 y2 ... xn yn <query_k1> q1 a1 <query_k2> q2 a2 ... <eos>
```

Here, $2n$ tokens encode n directed edges $x_i \to y_i$, forming a random permutation of n nodes.

The dataset is controlled by two parameters: N, the maximum permutation size, and K, the maximum reasoning depth. During training, n is sampled from $[3, N]$, while $k \in [1, K]$. Context lengths are fixed to 2048 tokens. We employ two variants of DEPO:

- DEPO1: Each node spans 1–2 tokens from vocab size 50, with $N = 225, 300, 375$ and $K = 8$.
- DEPO2: Each node spans 5–7 tokens from vocab size 4, with $N = 75, 100, 125$ and $K = 16$.

Evaluation focuses on both the hardest cases ($n = N$, $k = K$) and intermediate difficulty ($k = K/2$). For weaker models, we utilize *reduced* training setups with $K = 4$, denoted DEPO1$(K = 4)$ and DEPO2$(K = 4)$. The full methodological details are provided in Appendix A.1.

Task BREVO: Mental reasoning breadth. This evaluates a model's ability to process multiple dependencies simultaneously, as required in tasks involving tree-like traversal or dependency graphs. For example, solving queries like "Who are Alice's nephews?" or GSM-like examples requires parallel reasoning across branches of a graph to process relationships bottom-up [72]. Task BREVO isolates this capability using recursive traversal of directed acyclic graphs (DAGs), abstracting away natural language or arithmetic complexities. Each task instance is formatted as:

```
<bos> x1 y1 x2 y2 ... xm ym <query> q <ans> a1 a2 ... ap <eos>
```

Here, $2m$ tokens define m edges $x_i \to y_i$, representing dependencies where y_i depends on x_i. Upon receiving a query vertex q, the model outputs all vertices recursively reachable from q, sorted in topological order starting from the leaves (e.g., $u \to v \to q$ yields output u followed by v).

[3] This is observed in methods like ALiBi [44], Halibi [30], and Mimetic initialization [63], whose performance degrades on shorter contexts, as we show in this paper.

[4] Using CoT would reduce the k-hop task to simpler 1-hop associative recall.

The dataset is parameterized by N, the maximum graph size, with DAGs created using $n \leq N$ nodes, each of degree at most 4. Pretraining data is sampled by varying graph sizes, while testing focuses on the hardest graphs ($n = N$). We employ two variants of BREVO:

- BREVO1: Each vertex name spans a single token, with $N = 70/90/110$, fit within 1024 tokens.
- BREVO2: Name spans 2–4 tokens of vocab size 4, with $N = 30/40/50$, fit within 1536 tokens.

A key discovery from [72] revealed that, due to the non-uniqueness of valid outputs, language models must preprocess the entire topological order of the DAG *mentally* before generating the first token a_1. This insight confirms that our synthetic data rigorously evaluates reasoning breadth by requiring models to globally process the underlying graph structure before producing outputs.

Task CAPO: Knowledge capacity. Task CAPO evaluates a model's efficiency in encoding factual knowledge directly within its parameters, quantified as *bits per parameter*, which measures reliable storage capacity. Following the framework in [7], synthetic datasets of (fake) biographies are constructed to test knowledge retention. Each biography includes several attributes (e.g., birthdate, university, employer, etc.) and is presented in diverse paraphrased formats to reduce surface-level memorization [5, 6]. Capacity is measured using the next-token prediction distribution, accounting for both exact correctness and partial accuracy.

To highlight architectural differences, we adopt an undertrained regime where each biography is exposed only 100 times during pretraining.[5] The dataset includes $N = 50$K to 2M biographies, encoding 2×10^6 to 10^8 total bits of information. Models of varying sizes are tested, and results are visualized via "bit vs. model size" plots. Additional details are provided in Appendix A.3.

Task MANO: Knowledge manipulation. Task MANO evaluates a distinct form of reasoning: the ability to manipulate stored knowledge internally, contrasting with in-context reasoning tasks like DEPO or BREVO. While those tasks focus on reasoning over external tokens, MANO requires models to retrieve factual knowledge embedded in their parameters and perform hierarchical computation entirely mentally. This combination of retrieval and reasoning makes knowledge manipulation uniquely challenging and a skill that must be learned during pretraining.[6]

To test this capability, MANO employs synthetic modular arithmetic expressions inspired by human mental computation, particularly small-number arithmetic like the 9×9 multiplication table. Models solve multi-step arithmetic problems without intermediate steps like Chain-of-Thought. For example, given: `<bos> + * a b - c d <ans>` the task requires evaluating $((a \times b) + (c - d)) \bmod 23$ for $\ell = 3$, where operands a, b, c, d are sampled uniformly from $[0, 22]$. Modular arithmetic provides the foundational factual knowledge (23×23 operation tables), while the task challenges hierarchical reasoning by recursively composing operations. Additional details are provided in Appendix A.4.

The dataset is parameterized by a maximum expression length L, with ℓ sampled uniformly from $[1, L]$. We prepare three MANO datasets across difficulty levels: $L = 10$, 13, and 16.

Task LANO: Hierarchical language structure. Task LANO evaluates structural reasoning over hierarchical relationships and long-range dependencies. Unlike DEPO, BREVO, and MANO, which rely on explicit key-value pairs (in-context or knowledge), LANO challenges models to infer implicit recursive structures across sequences and resolve global ambiguities within them.

To test this, LANO leverages synthetic datasets built from context-free grammars (CFGs). Training sequences consist of CFG-valid sentences separated by `<bos>` tokens. For example:

```
<bos> 3 3 2 2 1 ... 3 3 1 2 <bos> 1 2 3 3 1 ... 1 2 2 1 <bos> ...
```

CFGs are designed with token-level ambiguity, where local tokens (e.g., 1, 2, 3) provide insufficient information to directly infer their mapping to CFG rules. Resolving this requires dynamic programming to globally map the entire sequence to a valid recursive application of CFG rules, which must

[5] Exposing each biography 1000 times during pretraining diminishes architectural differences, as even transformers without MLP layers can achieve similar storage efficiency [7]. Uniform exposure ensures clean systematic comparisons while avoiding confounding effects tied to rare outliers and junk data [7].

[6] For instance, questions like "Was [name] born in an even or odd month?" or derived 2-hop queries such as "What is [name]'s sister's birthdate?" demand reasoning layers over stored knowledge. These skills cannot reliably emerge through supervised fine-tuning alone [6] and require development during pretraining or continued pretraining.

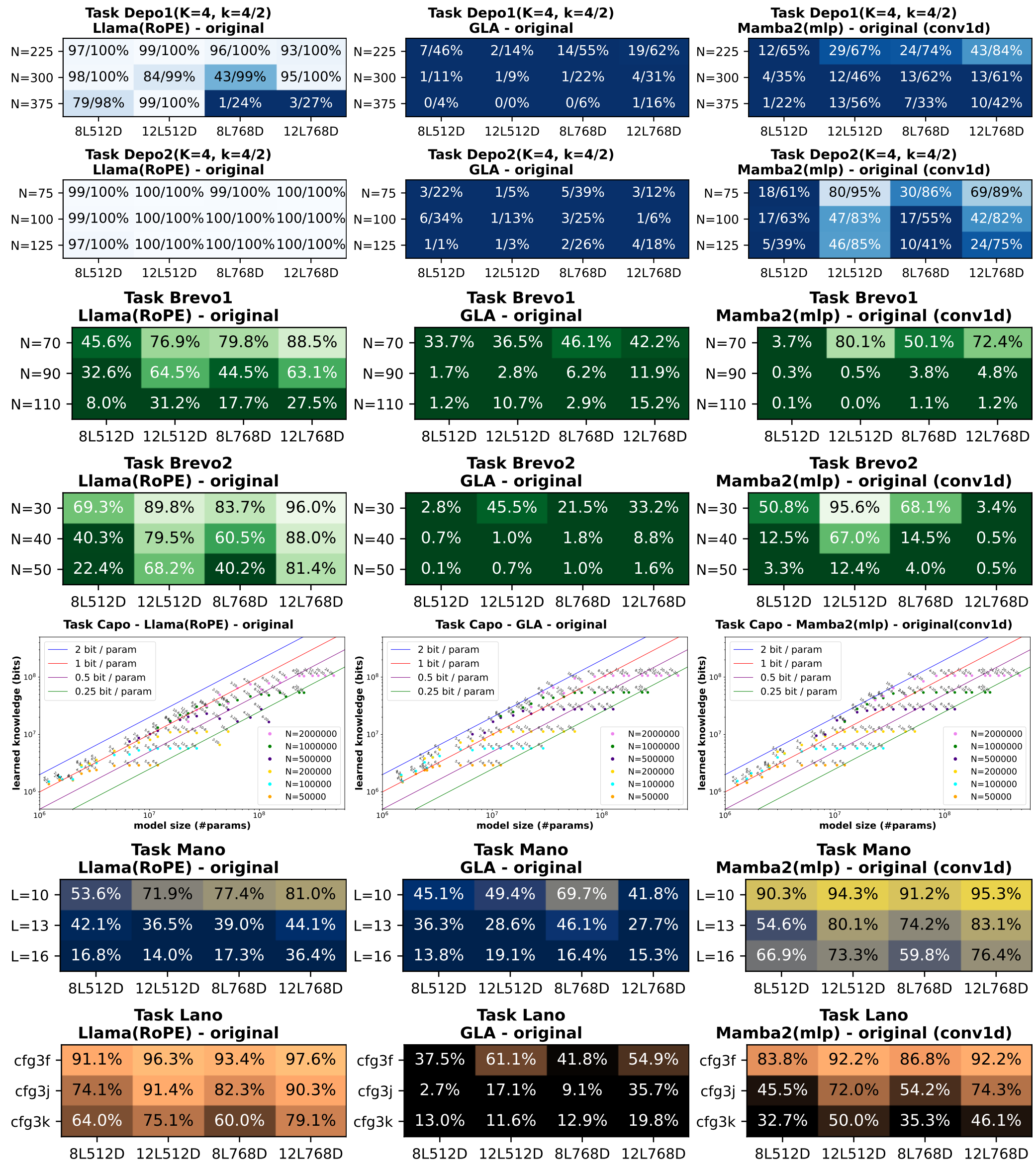

Figure 4: **Initial comparison** of RoPE, Mamba2, and GLA on five synthetic tasks. GLA performs poorly everywhere except knowledge capacity (CAPO); Mamba2 excels at knowledge (CAPO, MANO); Llama(RoPE) is best at reasoning (DEPO, BREVO, LANO). This confirms our synthetic playground **as effective** for architectural comparisons, but introducing Canon layers (see rest of the paper) will build a Pisa tower for more controlled and fair comparisons, where **the landscape shifts drastically** and reasoning depth improves 2–4×.

also be learned during training. This reasoning grows in worst-case complexity ($O(n^3)$) as sequence lengths increase. Details are in Appendix A.5.

Building upon cfg3f [3], which includes sequences of lengths 100–500, we introduce extended datasets cfg3j and cfg3k, with sequences ranging up to 200–1000 tokens to increase recursive depth and test models on more nested rules and longer dependencies. Training uses context lengths of 1536 for cfg3j and cfg3k, compared to 512 for cfg3f. Evaluation prompts models with <bos> to generate CFG-valid sentences, validated via a dynamic programming parser. KL divergence is also used to compare token distributions against ground truth.

In summary, this set of five synthetic tasks covers non-overlapping skills and distinct aspects of accuracy—token-level (DEPO, MANO), generative (BREVO, LANO), and distributional (CAPO, LANO). While this pool can be further enriched, it serves as a strong starting point for deriving meaningful architectural insights, as demonstrated in the following sections.

3 Initial Comparison on Well-Known Architectures

Language model architectures have evolved significantly since Transformers [64], resulting in three major families distinguished by computational mechanisms.

Quadratic-time attention models, pioneered by the original Transformer, include prominent architectures such as BERT [35] and GPT2 [46]. Recent refinements include Rotary Position Embeddings (RoPE) [12, 59] and gated MLP layers [54]. We use the Huggingface implementation of Llama, denoted as Llama(RoPE), incorporating RoPE and gated MLP, and a variant without positional embeddings, Llama(NoPE). We refer to these as RoPE and NoPE respectively when clear from the context. We exclude relative positional embeddings due to limited empirical benefits but additional computational costs [3].

RoPE models often generalize poorly beyond training context lengths. In contrast, NoPE generalizes better but suffers from lower overall performance. Recent attention-score modifications (e.g., ALiBi [44] and Hard-Alibi [30]) partially address this trade-off; we discuss in later sections.

Linear-time attention reduces computation by compressing sequences into fixed-length representations. Examples include Linformer [65], Performer [14], Linear Transformer [34]. We focus on more recent Gated Linear Attention (GLA) [69], known for computational efficiency and scalability.

Recurrent and state-space models process long sequences using evolving hidden states instead of attending over all tokens. Mamba [19, 26] exemplifies this category; we analyze its second generation (Mamba2). Other prominent models include S4 [56], S5 [56], RetNet [60], RWKV [42], HGRN [45], GSA [77], DeltaNet [71], and GatedDeltaNet [70].

Avoidance of hybrid architectures. We exclude models integrating attention with linear or state-space methods—e.g., Griffin [20], Samba [48], GatedDeltaNet-H1/H2 [70] or sliding-window attention—to maintain clarity. Such hybrid approaches excel in extremely long contexts (e.g., 1 million tokens), but our analysis focuses explicitly on precision within standard context windows (4096 tokens). In practice, long contexts are often compressed to shorter segments (e.g., via CoTs) for final detailed processing, making precise local reasoning essential.

Hybrid models can *obscure architectural trade-offs*; aggregated results may not reflect individual component contributions clearly. For instance, Mamba2 is strong in memory tasks yet weaker in structured reasoning. Hybrids blending linear/state-space modules with attention can mask these distinctions. Thus, for transparency, this study focuses entirely on isolated architectures to clearly analyze their inherent strengths and weaknesses.

Architecture Size Standardization. To ensure fair comparisons, we standardize model sizes and evaluate Llama, GLA, and Mamba2 as representative architectures from each family.

For all tasks except CAPO, we experiment with four architecture sizes. Llama models have 12 or 8 layers, with hidden dimensions of 768 or 512 (and 12 or 8 heads), denoted as `12L768D`, `8L512D`, etc. (`12L768D` matches GPT2-small). We translate these configurations into GLA, Mamba2, Mamba2(mlp) and Gated DeltaNet (GDN) to ensure comparable parameter counts.[7]

For CAPO (bit-per-parameter knowledge capacity), we vary model and data sizes more widely. Following [7], we denote model scale by ℓ-h: for Llama, this means ℓ layers, hidden size $64h$, and h heads. We extend this notation consistently to GLA and Mamba2.

Training. We use identical training settings (batch size, training steps, learning rates, etc.) across architectures to ensure fair comparisons. Complete details are provided in Appendix A. We also fix random seeds so that all architectures pre-train on precisely identical data sequences.

3.1 Initial Comparison Results

From Figure 4, linear-attention GLA performs weakest overall, Mamba2 excels in knowledge tasks (CAPO, MANO), and Llama(RoPE) performs best on reasoning tasks (DEPO, BREVO, LANO). These results validate the effectiveness of our synthetic playground; however, we avoid deeper interpretation at this point. As shown later, Llama and GLA lack a critical architectural component,

[7] The original Mamba2 has no MLP layers: each Mamba layer has $6d^2$ parameters (for hidden size d), compared with $12d^2$ in Llama. Thus, we configure Mamba2 with 24 or 16 layers to match Llama's size. Mamba2(mlp) alternates Mamba and gated MLP blocks, thus keeping 12 or 8 total layers. See details in Appendix C.

making this initial comparison incomplete, unfair, and less informative.

For now, we highlight several *key remarks*.

3×4 mini scaling laws. Randomness may affect outcomes. For example, in Task MANO, despite two seeds and four learning rates per configuration, smaller models sometimes outperform larger ones. Thus, robust statistical comparisons are crucial. We address this by testing our synthetic tasks systematically at *three* data scales and *four* architecture sizes (even more for Task CAPO). These "3×4" mini scaling laws enable clearer visual comparisons, reducing variability.

Benefits of synthetic tasks. Synthetic tasks clarify architectural differences starkly (e.g., 90% vs 5%), clearly exposing strengths and weaknesses. By contrast, real-world experiments often produce modest differences (e.g., 2%) buried in noise. Thus, synthetic pretraining environments allow clean evaluations of architectures' scalability and true capabilities.

Interpreting task failures. If a specific architecture (of a given size) fails at a certain difficulty level (e.g., large N or k), it does not imply the model cannot learn the skill given infinite training. Our comparison uses a fixed, limited training budget: all architectures train for the same number of steps with identical data and shuffling, reporting best accuracy across multiple learning rates. Thus, results should be seen as differences in the *speed of skill acquisition*, not absolute capability.[8]

Predicting future pipelines. Synthetic tasks simulate idealized, high-quality pretraining conditions targeting core skills like multi-hop reasoning (DEPO). Unlike datasets such as FineWeb-edu or SlimPajama, which contain sparse reasoning examples obscured by simpler content, synthetic tasks highlight core capabilities. Currently, 100B-token pretraining fails even simplest 2-hop reasoning (Result 12). As training pipelines evolve—via improved data curation or RL-based post-training—synthetic tasks like DEPO may better predict models' potential and guide architectural choices.

The remainder of this paper is deferred to the full version.

Acknowledgements

ZA sincerely thanks Vahab Mirrokni for the invitation to the Yale workshop in October 2023, where this research was sparked through enlightening discussions with Vahab Mirrokni and Peilin Zhong. Canon layers build on the idea of uniform attention previously explored in joint work with Yuanzhi Li [3]. ZA thanks Alberto Alfarano for introducing the papers [30, 44, 63, 79], and the PyTorch scaled dot product attention function. At Meta, we extend our heartfelt gratitude to Lin Xiao and Kristin Lauter for their insightful discussions and unwavering supports, which made this research possible. Special thanks go to Wangzhi Dai, Sam Doud, Dinesh Kannappan, Niki Kim, Junjie Qian, Ammar Rizvi, Travis Seevers, and Stephen Hartken at Meta, as well as Abraham Leal from W&B; without their invaluable technical assistance, the experiments presented in this paper would not have been feasible. We are deeply grateful to Songlin Yang and Ali Behrouz for providing detailed instructions on replicating their academic-scale pretraining experiments, and Fangcheng Sun for many helpful conversations on architecture design in general.
Contribution statement. ZA proposed all ideas, conducted all investigations, implemented all code, performed all experiments, authored the entire manuscript, and managed all necessary compliance reviews and social promotions; the term *Canon Layers* was jointly conceived and designed with Xiaoli Xu.

References

[1] Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

[2] Zeyuan Allen-Zhu and Yuanzhi Li. Can SGD Learn Recurrent Neural Networks with Provable Generalization? In *NeurIPS*, 2019. Full version available at `http://arxiv.org/abs/1902.01028`.

[8] Faster learning is practically important—for example, a model ideally learns reasoning skills quicker than pure memorization. Similar observations arise in knowledge capacity tasks [7], where architectural differences vanish with ample training but become pronounced when training budgets are limited.

[3] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of Language Models: Part 1, Learning Hierarchical Language Structures. *ArXiv e-prints*, abs/2305.13673, May 2023. Full version available at `http://arxiv.org/abs/2305.13673`.

[4] Zeyuan Allen-Zhu and Yuanzhi Li. Backward feature correction: How deep learning performs deep learning. In *COLT*, 2023. Full version available at `http://arxiv.org/abs/2001.04413`.

[5] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of Language Models: Part 3.1, Knowledge Storage and Extraction. In *Proceedings of the 41st International Conference on Machine Learning*, ICML 2024, 2024. Full version available at `http://arxiv.org/abs/2309.14316`.

[6] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of Language Models: Part 3.2, Knowledge Manipulation. In *Proceedings of the 13th International Conference on Learning Representations*, ICLR 2025, 2025. Full version available at `http://arxiv.org/abs/2309.14402`.

[7] Zeyuan Allen-Zhu and Yuanzhi Li. Physics of Language Models: Part 3.3, Knowledge Capacity Scaling Laws. In *Proceedings of the 13th International Conference on Learning Representations*, ICLR 2025, 2025. Full version available at `http://arxiv.org/abs/2404.05405`.

[8] Simran Arora, Aman Timalsina, Aaryan Singhal, Benjamin Spector, Sabri Eyuboglu, Xinyi Zhao, Ashish Rao, Atri Rudra, and Christopher Ré. Just read twice: closing the recall gap for recurrent language models. *arXiv preprint arXiv:2407.05483*, 2024.

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

[10] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[11] Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

[12] Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of the ACL Workshop on Challenges & Perspectives in Creating Large Language Models*, 2022. URL `https://arxiv.org/abs/2204.06745`.

[13] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[14] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

[15] 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.

[16] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2924–2936, 2019. doi: 10.18653/v1/N19-1300.

[17] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[18] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[19] Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms

through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024. URL https://arxiv.org/abs/2405.21060.

[20] 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.

[21] Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1246. URL https://aclanthology.org/N19-1246/.

[22] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.

[23] Daniel Y Fu, Tri Dao, Khaled Kamal 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. URL https://arxiv.org/abs/2212.14052.

[24] 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. A framework for few-shot language model evaluation, 07 2024. URL https://zenodo.org/records/12608602.

[25] Olga Golovneva, Tianlu Wang, Jason Weston, and Sainbayar Sukhbaatar. Multi-token attention. *arXiv preprint arXiv:2504.00927*, 2025.

[26] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. URL https://arxiv.org/abs/2312.00752.

[27] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv preprint arXiv:2005.08100*, 2020.

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

[29] Changho Hwang, Wei Cui, Yifan Xiong, Ziyue Yang, Ze Liu, Han Hu, Zilong Wang, Rafael Salas, Jithin Jose, Prabhat Ram, Joe Chau, Peng Cheng, Fan Yang, Mao Yang, and Yongqiang Xiong. Tutel: Adaptive mixture-of-experts at scale. *CoRR*, abs/2206.03382, June 2022. URL https://arxiv.org/pdf/2206.03382.pdf.

[30] Samy Jelassi, David Brandfonbrener, Sham M Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. *arXiv preprint arXiv:2402.01032*, 2024.

[31] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023. doi: 10.1145/3571730. URL https://doi.org/10.1145/3571730.

[32] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[33] Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

[34] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International conference*

on machine learning, pages 5156–5165. PMLR, 2020.

[35] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[36] 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.

[37] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL `https://aclanthology.org/Q19-1026/`.

[38] Nayoung Lee, Ziyang Cai, Avi Schwarzschild, Kangwook Lee, and Dimitris Papailiopoulos. Self-improving transformers overcome easy-to-hard and length generalization challenges. *arXiv preprint arXiv:2502.01612*, 2025. URL `https://arxiv.org/abs/2502.01612`.

[39] OpenAI. Gpt-4 technical report, 2023.

[40] Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, 2016. doi: 10.18653/v1/P16-1144.

[41] Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL `https://arxiv.org/abs/2406.17557`.

[42] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

[43] Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022. URL `https://arxiv.org/abs/2201.02177`.

[44] 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.

[45] Zhen Qin, Songlin Yang, and Yiran Zhong. Hierarchically gated recurrent neural network for sequence modeling. *Advances in Neural Information Processing Systems*, 36:33202–33221, 2023.

[46] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[47] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia, 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124. URL `https://aclanthology.org/P18-2124/`.

[48] 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.

[49] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.

[50] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, 2019. doi: 10.18653/v1/D19-1454.

[51] Eshika Saxena, Alberto Alfarano, Emily Wenger, and Kristin Lauter. Teaching transformers modular arithmetic at scale. *arXiv preprint arXiv:2410.03569*, 2024. URL `https://arxiv.org/abs/2410.03569`.

[52] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[53] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[54] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[55] 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. In *International Conference on Learning Representations*, 2016.

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

[57] DR So, W Manke, H Liu, Z Dai, N Shazeer, and QV Le. Primer: Searching for efficient transformers for language modeling. arxiv 2021. *arXiv preprint arXiv:2109.08668*.

[58] Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama. `https://www.cerebras.net/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama`, 2023. URL `https://huggingface.co/datasets/cerebras/SlimPajama-627B`.

[59] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2021.

[60] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

[61] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[62] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[63] Asher Trockman, Hrayr Harutyunyan, J Zico Kolter, Sanjiv Kumar, and Srinadh Bhojanapalli. Mimetic initialization helps state space models learn to recall. *arXiv preprint arXiv:2410.11135*, 2024.

[64] 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.

[65] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[66] Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart Van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.

[67] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22–31, 2021.

[68] Songlin Yang and Yu Zhang. Fla: A triton-based library for hardware-efficient implementations of linear attention mechanism, January 2024. URL `https://github.com/fla-org/flash-linear-attention`.

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

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

[71] Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear

transformers with the delta rule over sequence length. *arXiv preprint arXiv:2406.06484*, 2024.

[72] Tian Ye, Zicheng Xu, Yuanzhi Li, and Zeyuan Allen-Zhu. Physics of Language Models: Part 2.1, Grade-School Math and the Hidden Reasoning Process. In *Proceedings of the 13th International Conference on Learning Representations*, ICLR 2025, 2025. Full version available at `https://arxiv.org/abs/2407.20311`.

[73] Tian Ye, Zicheng Xu, Yuanzhi Li, and Zeyuan Allen-Zhu. Physics of Language Models: Part 2.2, How to Learn From Mistakes on Grade-School Math Problems. In *Proceedings of the 13th International Conference on Learning Representations*, ICLR 2025, 2025. Full version available at `http://arxiv.org/abs/2408.16293`.

[74] Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. Distilling system 2 into system 1. *arXiv preprint arXiv:2407.06023*, 2024.

[75] Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, YX Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. *arXiv preprint arXiv:2502.11089*, 2025.

[76] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, 2019. doi: 10.18653/v1/P19-1472.

[77] Yu Zhang, Songlin Yang, Rui-Jie Zhu, Yue Zhang, Leyang Cui, Yiqiao Wang, Bolun Wang, Freda Shi, Bailin Wang, Wei Bi, et al. Gated slot attention for efficient linear-time sequence modeling. *Advances in Neural Information Processing Systems*, 37:116870–116898, 2024.

[78] Zhengyan Zhang, Yixin Song, Guanghui Yu, Xu Han, Yankai Lin, Chaojun Xiao, Chenyang Song, Zhiyuan Liu, Zeyu Mi, and Maosong Sun. Relu^2 wins: Discovering efficient activation functions for sparse llms. *arXiv preprint arXiv:2402.03804*, 2024.

[79] 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.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions—Canon layers and a synthetic pretraining framework—and these are supported throughout the paper with 12 results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper notes that academic-scale pretraining introduces noise and that the current five-task synthetic suite, while sufficient for distinguishing base models, is not exhaustive.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: The paper does not present formal theorems or proofs, and is focused on empirical architectural evaluation.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Detailed training protocols, hyperparameters, data generation, and evaluation procedures are provided in the appendix of the full version of this paper.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The code and data generators are not yet released, but we plan to open-source them soon; at this point, we have tried to provide all the technical details needed for reproducibility.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Full experimental configurations are described in Appendix A, B and B, including model sizes, optimizers, learning rates, batch sizes, and evaluation steps.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The paper quantifies noise due to random seeds (e.g., Figure 1) and only interprets performance differences above that margin.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper specifies hardware (e.g., A100/H100 GPUs), training duration, batch sizes, and Canon layer runtime overhead (Appendix 4 and A).

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses only synthetic or publicly available datasets and does not involve human subjects, privacy risks, or misuse scenarios.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

   Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

   Answer: [Yes]

   Justification: The Conclusion discusses potential benefits of more reproducible and resource-efficient architecture evaluations, and no foreseeable negative impacts are identified.

   Guidelines:

   - The answer NA means that there is no societal impact of the work performed.
   - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
   - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
   - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
   - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
   - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper does not release high-risk models or data; it works with synthetic benchmarks and academic-scale training only.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All external models and datasets (e.g., Mamba2, SlimPajama) are cited with source, version, and licenses are respected.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [No]

Justification: New synthetic datasets and evaluation suites are introduced, but not yet released; documentation is provided in Appendix A and release is planned post-submission.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve any human subjects or crowdsourcing.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects were involved, and no IRB approval was necessary.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [NA]

    Justification: LLMs were not used as an important, original, or non-standard component of the core research methods described in this paper. Any LLM usage was for general writing assistance.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.