

Fast Compression versus Exact Recall: Investigating the Trade-offs Between Models in Specialized Reasoning Tasks

Stanford CS224N Custom Project

Jerry Xiao

Department of Computer Science
Stanford University
jxiao45@stanford.edu

Nick Yan

Department of Computer Science
Stanford University
nyan24@stanford.edu

Abstract

Transformers offer explicit token-to-token retrieval attention but incur quadratic cost in sequence length, while Mamba scales linearly and uses recurrent state updates that compress past context. We investigate this trade-off by evaluating GPT-2 (Transformer), Mamba-130m (SSM), and a Hybrid Mamba-Transformer model on two types of tasks: long-context literature continuation and mathematical reasoning. On the literature task (Gutenberg, 7000-token context), we find that Mamba outperforms GPT-2, supporting the hypothesis that selective compression benefits synthesis-oriented tasks. On the mathematical/analytical reasoning tasks (OlympiadBench, GSM8K, AMC23), the Transformer consistently outperforms Mamba, suggesting that tasks requiring exact recall favor the attention mechanism. A lightly pretrained hybrid model outperforms GPT-2 on literature perplexity (29.39 vs. 29.61), matches GPT-2 on AMC23 (90%), and falls between the GPT-2 and Mamba baselines on OlympiadBench and GSM8K. The results suggest that task-architecture alignment may matter more than pursuing a single universal architecture, and our interpretability probes provide preliminary evidence to differentiate these retrieval and compression behaviors.

1 Key Information to include

- Mentor: Wu Fang
- External Collaborators (if you have any): N/A

2 Introduction

When performing tasks, should the model compress the text and store the most important context or should it have precise recall over all past words and their relation to their current word? The answer varies across tasks and architectures. Transformers are widely used for language modeling tasks, but their self-attention mechanism relies on a key-value cache that scales quadratically with sequence length. Mamba offers an alternative with linear-time context compression through selective state-space models, but lacks exact attention recall mechanisms.

The main objective of this project is to test whether models' architectural differences in context handling lead to different strengths across different tasks. In particular, we hypothesize that the Transformer performs better when success depends on the precise retrieval of earlier token-level information (e.g. mathematical reasoning tasks, where small details and intermediate steps are crucial), while Mamba performs better when success depends on compressing long contexts into a coherent summary. The motivation for pursuing this task is not only to gain a deeper understanding of how different model architectures influence their understanding of different contexts, but to see

how to flexibly adapt model architectures to specialize within different tasks and see if a generalized hybrid architecture that can perform well across many forms of specialized reasoning.

To give a brief analogy, consider the two models attempting to read a book. The key-value caching system under the Transformer’s self-attention mechanism is akin to photographing every single page and memorizing it. On the other hand, Mamba’s recurrent state space lets it read fast and take brief notes at the expense of not recalling exact details that the Transformer might remember, such as the first word on the fourth page. We also investigate whether it is possible to build a Hybrid Mamba-Transformer model that combines these two behaviors, potentially improving long-context synthesis without fully sacrificing the contextual awareness of past details.

To investigate which kinds of tasks each architecture may outperform the others in, we use two different evaluation setups: synthesis-based tasks which require contextual awareness and analytical tasks which require careful, step-by-step reasoning. For the synthesis-based task, we compare GPT-2 (Transformer), Mamba-130m (SSM), and a Hybrid Mamba-Transformer across Gutenberg literature samples, where we measure perplexity on 1192-token continuations given 7000 token contexts for each text. For the analytical task, we use three mathematical evaluation benchmarks of varying difficulty: OlympiadBench, AMC23, and GSM8K. More detail on the set-up will be offered in the following sections.

Our results support the hypothesis: Mamba achieves substantially lower perplexity than GPT-2 on literature, while the Transformer consistently outperforms Mamba on mathematics, with the gap largest on AMC23 (90% vs. 40%). The hybrid model improves GPT-2’s perplexity while also keeping its performance on mathematical tasks, improving upon the lower accuracy from Mamba. However, the results still suggest that it is more important for architectures to align with tasks rather than finding a universal hybrid model that outperforms both architectures.

3 Related Work

In “Language Models are Unsupervised Multitask Learners”, Radford et al. [1] introduced GPT-2, making the claim that language modeling, at a sufficient scale, can perform downstream tasks without explicit supervision. A model trained purely to predict the next token on a diverse corpus such as WebText therefore learns to infer and perform many tasks from naturally occurring demonstrations. What enables this behavior is the Transformer’s self-attention mechanism: by maintaining a key-value cache over the entire context, each token can attend to any previous token with content-based relevance. This design supports both learning from long documents where task demonstrations span many tokens and performing tasks that require precise recall of facts. By refusing to forget and maintaining explicit storage and retrieval of all context, the Transformer can generalize in this way, at the cost of quadratic scaling in sequence length.

Prior to the invention of Mamba, researchers sought many ways to achieve sub-quadratic complexity in sequence modeling. Arjovsky et al. [2] proposed Unitary Evolution Recurrent Neural Networks. Their unitary matrix method was able to parametrize these unitary matrices through simple constructions (diagonal phases, reflections, permutations) that achieved $O(n \log n)$ computation and $O(n)$ memory per step. The uRNN preserves the information rather than compressing it, with its hidden state retaining information over a long span. Separately, Gu et al. [3] introduced HiPPO, which provides a principled framework for how State Space Models compress history. By framing memory as a function approximation problem, where past signal is projected onto polynomial bases, HiPPO produced a recurrent update that scales through time to approximate all previous inputs. Thus, SSMs are able to compress history into a compact state, though initially ill-suited for natural language filtering because standard SSMs apply the same recurrent transformation to every token, without content-dependent filtering.

By creating Mamba, Gu and Dao [4] adapted SSMs for language modeling. Their key innovation was the selection mechanism: to address the fixed recurrence problem, Mamba parametrizes the discretization so that the model can actively decide what to propagate, compress, or discard at each step. This allows the architecture to filter irrelevant information and retain a compressed summary of the most important context, achieving linear-time scaling while processing text and matching or exceeding Transformer performance at comparable scale. By using a GPU-efficient algorithm that avoids the bottlenecks of recurrent scanning, they were able to use lower memory and compute.

However, Mamba’s input-dependent selection actively forgets and compresses, which helps for compressing discrete data but may struggle on tasks requiring exact recall.

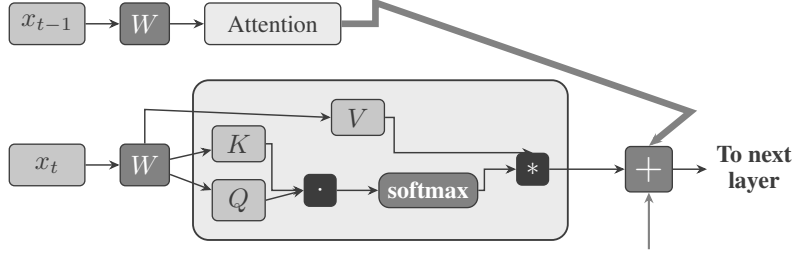
4 Approach

We compare three model families across synthesis-based and analytical tasks: a Transformer (GPT-2), a selective state-space model (Mamba-130m), and a hybrid architecture that utilizes both Mamba and Transformer attention. As our baselines, we use GPT-2 and Mamba-130m via HuggingFace:

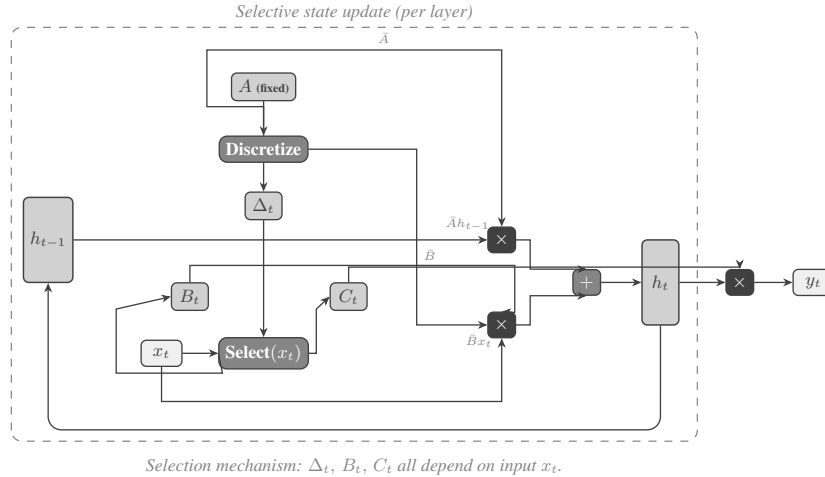
- GPT-2 (117M parameters, 12 transformer layers) is loaded from gpt2 [5]. This model only uses attention blocks, which use scaled dot-product self-attention:

$$\text{Attn}(Q, K, V) = \text{softmax}(QK^\top / \sqrt{d})V$$

with a causal mask. The key-value cache provides full pairwise access over the context. Below is the diagram for the GPT-2 self-attention mechanism: input x_t is projected via W into keys K , queries Q , and values V . The dot product $K \cdot Q$ is passed through softmax to produce attention weights, which scale V before being added to the residual stream.



- Mamba-130m (130M parameters, 24 SSM layers): This model is loaded from state-spaces/mamba-130m-hf [4]. Each layer uses the S6 selective scan, which compresses past context into a recurrent state through input-dependent discretization. To ensure compatibility, both baselines use their respective BPE tokenizers throughout all experiments. [6] In the SSM, the input parameter x_t is passed in and updated with hidden state $h_t = \bar{A}h_{t-1} + \bar{B}x_t$ where $\bar{A}$ and $\bar{B}$ are discretized from Δ_t . Then $y_t = C_t h_t$ is sent into the next layer with h_t fed back as the recurrent state (compressing all past context).

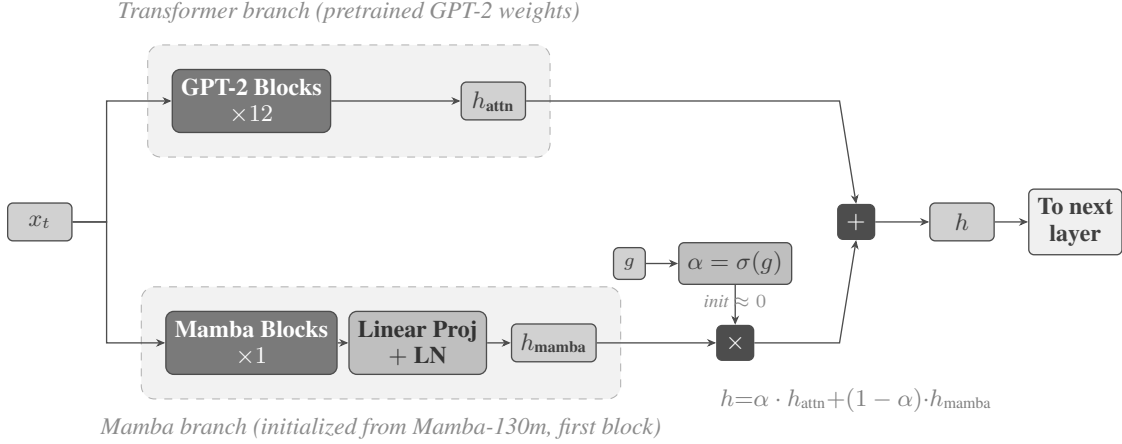


The main hybrid we will build and evaluate in this experiment is the Hybrid Mamba-Transformer (HMT), which runs a Transformer branch and a Mamba branch in parallel before combining their outputs. This design reflects our hypothesis; if attention and recurrent compression capture complementary information, then perhaps a gated model combining these branches could exploit both

signals. The Transformer branch consists of 12 GPT-2 blocks and the Mamba branch consists of one Mamba block followed by a linear projection and layer norm. The combined hidden state is

$$h = \alpha \cdot h_{\text{attn}} + (1 - \alpha) \cdot h_{\text{mamba}}, \quad \alpha = \sigma(g)$$

where $g \in \mathbb{R}$ is a learned gate (initialized to zero so $\alpha = 0.5$). We initialize the Transformer branch, embeddings, and LM head from pretrained GPT-2, and the Mamba blocks from Mamba-130m.



4.1 Pretrained Initialization

For the hybrid model, we initialize the Mamba branch from the first block of Mamba-130m, using a fusion linear layer that allows the Mamba features to pass through. The HMT copies GPT-2’s token embedding, positional embedding, 12 attention blocks, final layer norm, and LM head. To keep the training budget modest, we freeze the pretrained GPT-2 branch and train only the Mamba branch and fusion gate for 500 steps. This setup lets us test whether a small amount of additional capacity can improve task alignment without requiring full-on training.

5 Experiments

5.1 Data

Our experiments used a long-context literature corpus (synthesis-based task) and three mathematical reasoning corpora (analytical task).

For literature, we use the `gutenberg-en-v1-clean` dataset. The processing pipeline only selects texts with scores greater than 0.95, randomly samples 500 books, tokenizes them with the GPT-2 tokenizer, and constructs long-context evaluation examples consisting of a 7000-token context window and 1192-token target. [7]

Mathematical reasoning was tested on three datasets in increasing order of difficulty: GSM8K, which covers grade-school level word problems; AMC23, which covers high-school level competition math; and the English-language split of OlympiadBench, which covers Olympiad-level mathematics. The processing pipeline keeps the question, solution, final answer, and subfield fields, while normalizing LaTeX syntax to reduce formatting variance during training and evaluation. [8] [9] [10]

5.2 Evaluation method

We used two primary evaluation metrics. For the synthesis-based task (literature), we calculate perplexity, the next-token cross-entropy loss, over the 1192 target tokens given the preceding context.

$$\text{Perplexity} = \exp \left(-\frac{1}{N} \sum_{i=1}^N \log P(x_i | x_{<i}) \right)$$

Lower perplexity indicates a better result since it shows the model is more confident about its prediction. If the model’s maximum context length is under the 7000-token limit, we use the most recent tokens so that the target tokens receive the relevant preceding text. To compute the overall result across all samples, we compute the mean negative log-likelihood over the target tokens conditioned on the context, where context is truncated to the model’s maximum position length when necessary.

For our analytical tasks, we evaluate on three different mathematical reasoning benchmarks: OlympiadBench, AMC23, and GSM8K, and we assess whether the model produced an exact match via a greedy decoding of 64 new tokens from the question prompt. We prompt the models with `Question: {question}\n\nFinal answer:` and use a 512-token context window for the question, stopping decoding at end-of-sequence or after 64 generated tokens. A prediction counts as correct if the original normalized answer string matches the normalized prediction string. Therefore, a higher score is better, as it indicates a higher accuracy. This metric is simple and reproducible, and we complement this with an error taxonomy later in the paper.

5.3 Experimental details

GPT-2 and Mamba were evaluated in their pretrained zero-shot form, while the HMT received a small amount of task-specific adaptation (500 steps on mixed math and literature; batch size 2 with the GPT-2 branch frozen). This makes the hybrid experiment a lightweight adaptation study rather than a full end-to-end hybrid training run, and the resulting comparisons should be interpreted accordingly.

We used a context window of 1024 tokens, which is GPT-2’s hard position-embedding limit. The HMT used pretrained weight initialization as described above. For literature/perplexity evaluation, we cap evaluation to 50 samples and truncate the scored portion to 512 tokens. For analytical evaluation, we evaluate 100 samples from OlympiadBench, 100 samples from GSM8K, and 10 examples from AMC23 (due to the limited size of the dataset), and we generate up to 64 new tokens per problem.

The default training configuration uses batch size 2, maximum sequence length 512, AdamW optimization, 100 warmup steps, gradient clipping at 1.0, and 500 training steps for the hybrid model.

5.4 Results

	GPT-2	Mamba-130m	HMT
Perplexity (literature)	29.61	20.70	29.39
Exact match (OlympiadBench)	0.09	0.07	0.08
Exact match (AMC23)	0.9	0.4	0.9
Exact match (GSM8K)	0.08	0.06	0.07

The main observation is that Mamba performs best on the long-context literature task (lowest perplexity) while GPT-2 performs best on the mathematical reasoning tasks (highest accuracy). The HMT’s performance are consistent with our hypothesis; it slightly outperforms GPT-2 on literature and exhibits GPT-2-like behavior on math. Additional tables and graphs about these runs, such as context-length sensitivity measurements, hybrid gate analysis for the HMT, attention analysis, and token saliency have been listed in the analysis.

6 Analysis

To understand how the model’s architecture affected their final results, we ran analyses across all three models in addition to our evaluation.

As Figure 1 shows, the perplexity decreases as context length increases. Mamba consistently achieves the lowest perplexity across all lengths, while HMT remains slightly better than GPT-2. This suggests that all models benefit from additional context, yet Mamba is especially effective at converting long contexts into useful summary information for literature continuation. On the other hand, Transformers may allocate more computation to explicit token-level retrieval; overall, literature continuation seems to reward effective summary representations more than the exact recall of individual earlier tokens.

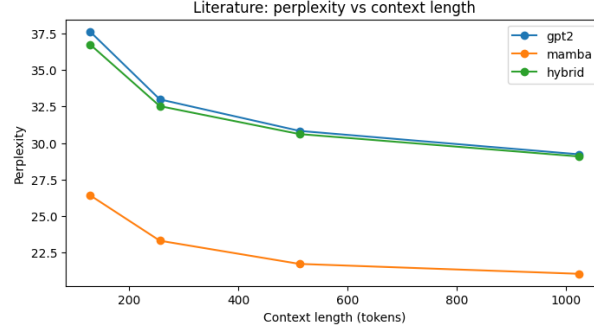


Figure 1: Literature Performance: Perplexity Versus Context Length

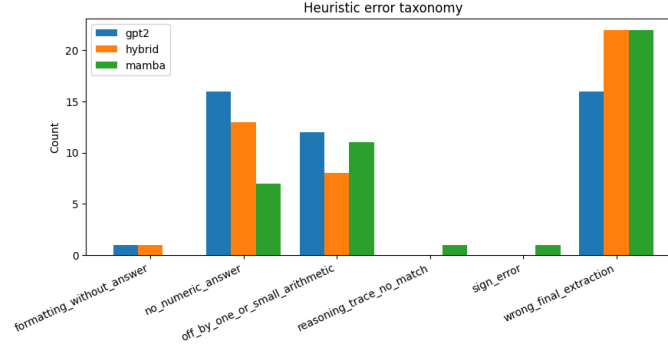


Figure 2: Error Taxonomy

Figure 2 groups the mathematical task errors into heuristic categories. The most common error for all models is `wrong_final_extraction`, indicating that final-answer evaluation is influenced not only by the underlying computation but also by whether the model cleanly extracts the answer in the expected format. GPT-2 made the fewest extraction errors, though it produces more `no_numeric_answer` cases. This suggests that part of GPT-2’s advantage on the math benchmarks may come from its better preservation and retrieval of token-level details at the output stage. HMT inherits some of GPT-2’s behavior, which is consistent with its stronger math performance relative to Mamba.

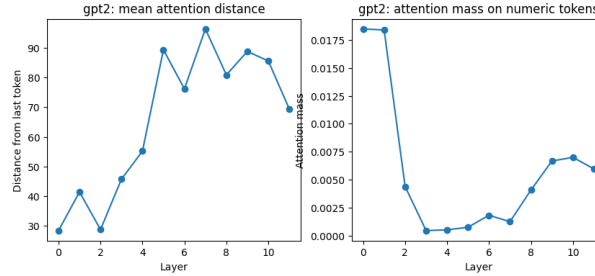


Figure 3: GPT-2 Attention Summary

In Figure 3, we analyze GPT-2’s attention: its mean attention distance from the final token and total attention mass assigned to numeric tokens. The mean attention distance measures the average number of positions between the last token and the tokens it attends to, where higher values indicate longer-range attention. The attention mass on numeric tokens is measured by the fraction of total attention weight assigned to tokens that contain digits, such as numbers in math problems. Our plot indicates that the distance peaks in the middle layers (around Layer 7), which show long-range attention, while the attention mass on numeric tokens is high in early layers but decreases in the

middle before rising again in deeper layers. The plots suggest a consistent pattern: GPT-2 treats numbers as salient anchor tokens and integrates them with context in the intermediate layers before re-attending to them for the final output. This retrieval-oriented behavior aligns with GPT-2’s stronger performance on the math benchmarks.

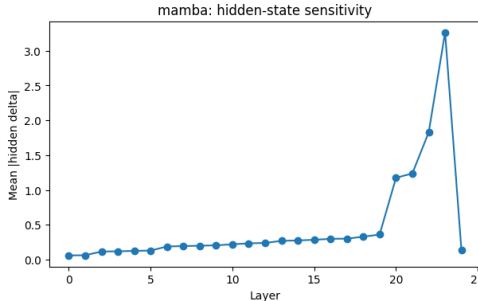


Figure 4: Mamba Hidden State Sensitivity: mean delta across layers

Figure 4 details Mamba’s hidden state sensitivity. Mamba does not expose an explicit token-to-token lookup like Transformers do, so this is a useful proxy for how much Mamba is preserving or updating its internal summary at each step. As such, small hidden-state deltas indicate the model is carrying forward a relatively stable summary of prior context, while large hidden-state deltas indicate stronger rewriting or updating of that context. Sensitivity remains relatively small through most layers of the network and rises sharply near the top of the network, especially layer 23. This suggests that Mamba preserves a compressed running representation across depth and performs stronger selective transformation only in later layers. This is consistent with our hypothesis of Mamba functioning more as a summary mechanism rather than an explicit token-retrieval mechanism, thus explaining its strength on synthesis-oriented literature tasks.

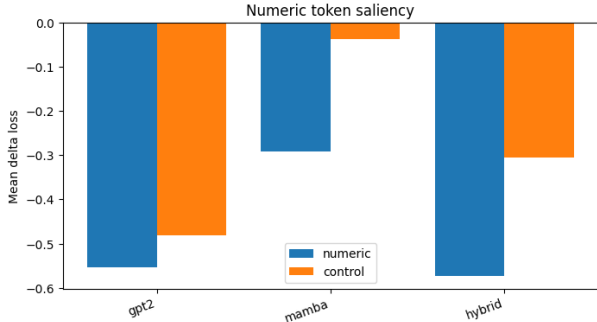


Figure 5: Numeric Token Saliency

In Figure 5, we measure how the model’s loss changes with numeric versus non-numeric tokens in the prompt are ablated. GPT-2 and HMT are substantially more sensitive to numeric tokens, which indicates a greater dependence on exact numeric content during prediction. On the other hand, Mamba shows a smaller response, consistent with a model that relies less on preserving the identity of specific tokens. This also supports the hypothesis as the Transformer and HMT rely on precise token-level information (also taking into account positional embeddings), which help on math tasks that require closer attention to detail, while Mamba compresses and attenuates such details, favoring broader content instead.

7 Conclusion

We studied whether explicit attention-based retrieval and recurrent state compression lead to different strengths on different types of tasks. Indeed, GPT-2 performed better on mathematical reasoning tasks, which required precise recall of information gathered in intermediate steps, while Mamba performed

better on long-context literature continuation. These results show the importance of task-architecture alignment.

Our hybrid Mamba-Transformer partially bridged this gap. With modest additional training, it slightly improved GPT-2 on literature perplexity while remaining closer to GPT-2 than to Mamba on the math tasks. At the same time, the hybrid did not uniformly outperform both baselines, indicating that combining explicit retrieval and compressed recurrence remains nontrivial even when the two branches are fused directly.

The interpretability analyses are consistent with these behavioral differences. GPT-2 shows increasing long-range attention with depth and higher sensitivity to numeric tokens, both of which fit a retrieval-based account of symbolic reasoning. Mamba, by contrast, shows relatively stable hidden-state evolution through much of the network followed by stronger late-layer refinement, consistent with a compressed-summary mechanism for long-context processing.

This study is necessarily limited by the relatively small scale of the evaluation, and the hybrid only received lightweight adaptation. Future work should test these hypotheses on larger controlled benchmarks which match tokenizers and context length across all three architectures. Stronger hybrid designs should also be explored with adaptations of interleaved layers and training procedures that allow both branches to adapt jointly, instead of freezing one branch.

Additionally, more causal interpretability methods should be used to better isolate when retrieval-based versus compression-based memory is most useful. One possible hybrid variant that can be further explored is one in which Mamba is used to selectively filter and reduce the Transformer’s key-value cache by selecting those most relevant to the current context. This will allow the SSM’s compression aid the attention mechanism rather than replace it, which could reduce memory overhead while preserving the exact retrieval that math tasks require.

However, this hybrid variant raises ethical concerns. If the selective mechanism learns to discard information that is rare or statistically less frequent, it may also remove context that is important for representing minority perspectives and other socially meaningful details. This occurs because Mamba’s selective filter may erase low-frequency tokens and contexts that it deems irrelevant to synthesizing information, and ultimately risks amplifying the model’s existing biases. For this reason, future work should evaluate fairness and representation alongside efficiency and performance.

Team contributions (Required for multi-person team)

Jerry Xiao implemented the HMT model and the evaluations, loaded the GPT-2 and Mamba models, and ran the evaluations across all models. Nick Yan gathered and parsed the datasets, helped in implementing the HMT model, and wrote/ran the analysis code. Both contributed equally to the final paper.

References

- [1] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.
- [2] Shah A. Bengio Y Arjovsky, M. Unitary evolution recurrent neural networks. In *International Conference on Machine Learning (ICML)*, 2016.
- [3] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. <https://arxiv.org/abs/2008.07669>, 2020. arXiv:2008.07669.
- [4] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling (COLM)*, 2024.
- [5] openai-community. gpt2. <https://huggingface.co/openai-community/gpt2>, n.d. Hugging Face model repository.
- [6] state-spaces. mamba-130m-hf. <https://huggingface.co/state-spaces/mamba-130m-hf>, n.d. Hugging Face model repository.

- [7] BEE-spoke-data. `gutenberg-en-v1-clean`. <https://huggingface.co/datasets/BEE-spoke-data/gutenberg-en-v1-clean>, n.d. Hugging Face dataset repository.
- [8] OpenBMB. OlympiadBench. <https://github.com/OpenBMB/OlympiadBench>, 2024. GitHub repository (dataset and code).
- [9] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- [10] math-ai. AMC23 Dataset. <https://huggingface.co/datasets/math-ai/amc23>, 2025. Hugging Face dataset.